IPS Instituto Politécnico de Setúbal
Escola Superior de Tecnologia de Setúbal

Luís Miguel Varela

Segmentação temporal de vídeo em tempo real

Dissertação de Mestrado em Engenharia Eletrotécnica e Computadores, Ramo de Eléctrónica e Computadores

setembro de 2013

IPS Instituto Politécnico de Setúbal
Escola Superior de Tecnologia de Setúbal

Luís Miguel Varela

Segmentação temporal de vídeo em tempo real

Orientador:
Rui Pedro Batoreo Amaral

Co-orientador:
Tito Gerardo Batoreo Amaral

Dissertação de Mestrado em Engenharia Eletrotécnica e Computadores, Ramo de Eléctrónica e Computadores

setembro de 2013

em memória de João Varela

Agradecimentos

Nesta secção gostaria de expressar publicamente os meus sinceros agradecimentos a todas as pessoas que me acompanharam nesta etapa da minha vida e de alguma forma contribuíram para a realização e enriquecimento do trabalho apresentado nesta tese.

Em primeiro lugar agradeço à minha mãe, Maria Guilhermina Varela, que sozinha, e com pouco mais que o salário mínimo, me suportou até ao final da licenciatura. Sem ela, e sem os sacrifícios que fez, este trabalho nunca teria sido realizado.

Em seguida agradeço aos meus orientadores, Professor Doutor Rui Amaral e ao Professor Doutor Tito Amaral pela sua paciência, espírito crítico, dedicação e apoio. Posso dizer que me transmitiram não só conhecimentos técnicos e científicos, mas também, o gosto e curiosidade pelo processamento digital de vídeo e imagem.

E por último gostaria de agradecer a todos os meus amigos que, de uma maneira ou outra, contribuíram para a conclusão deste trabalho. Em especial à Laura Soraia Perpétuo pelas horas que perdeu a corrigir erros ortográficos e gramaticais.

Resumo

A segmentação temporal de vídeo é o processo de automaticamente detetar transições entre unidades estruturais de um vídeo (*shots*, cenas ou histórias). É um problema que tem atraído muita atenção por parte da comunidade cientifica pois é uma fase essencial de qualquer ferramenta de análise de vídeo.

Este trabalho descreve algoritmos de segmentação de vídeos de programas noticiosos por *shots* e por histórias desenvolvidos no âmbito no Mestrado em Engenharia Eletrotécnica, Ramo de Eletrónica e de Computadores. O Algoritmo de Segmentação por *Shots* consiste numa métrica calculada a partir da deteção de contornos e num limiar adaptativo. Por outro lado, o Algoritmo de Segmentação por Histórias é baseado na deteção de pivôs através da análise de histogramas do vestuário dos mesmos.

Palavras-chave: Programas Noticiosos, Processamento de Vídeo, Segmentação por *Shots*, Segmentação por Histórias.

Abstract

Temporal video segmentation is the process of automatically detecting transitions between structural units of a video (shots, scenes or stories). It is a problem that has drawn much attention from the scientific community because it is an essential tool for any video analysis.

This paper describes algorithms for the segmentation, at the shot and story level, of late evening news shows. This work was developed in order to obtain the Master's Degree on Electrical and Computer Engineering. The shot segmentation algorithm was based on a contour detection metric and an adaptive threshold. Moreover, the story segmentation algorithm detects news anchors by comparing the histogram of their clothing with an automatically selected template.

Key-words: News Shows, Video processing, Shot Segmentation, Story Segmentation.

Índice

Lista de Figuras

Lista de Tabelas

Lista de siglas e acrónimos

Fn	*False Negatives*
Fp	*False Positives*
FPS	*Frames Per Second*
GPGPU	*General-Purpose Computing on Graphics Processing Units*
LDA	*Latent Dirichelet Allocation*
NASA	National Aeronautics and Space Administration
NIST	National Institute of Standards and Technologies
PCA	*Principal Component Analysis*
Pre	*Precision*
QVGA	*Quarter Video Graphics Array*
Rec	*Recall*
RTP	Rádio e Televisão de Portugal
SCG	*Shot Connectivity Graph*
SIC	Sociedade Independente de Comunicação
SQL	*Structured Query Language*
SVM	*Support Vector Machine*
Tp	*True Positives*
VDB	*Visual Database*
XML	*Extensible Markup Language*

Lista de Símbolos

a Largura da face medida

af Área da face em relação à área da face da *template-pivô*

A Área da região

b Largura de ombros estimada

B Número de *bins* do espaço de cor

C Imagem de contornos

CD Característica de contornos

CD_{max} Valor máximo da característica de contornos

d Distância

dh Distância entre histogramas

f *Frame*

F_β *F-measure*

F_1 *F-measure* com β igual a 1

F_3 *F-measure* com β igual a 3

F_s Potencial início de transição

H Função histograma

i Imagem original

ii Imagem integral

L *Likelihood*

M Largura em píxeis

N Altura em píxeis

p_r Fração da imagem definida pela região

P Probabilidade

P_a Pivô ausente

P_p Pivô presente

r Região

R Número de regiões da imagem

s Semelhança

$S1$ Primeiro cenário

$S2$ Segundo cenário

$S3$ Terceiro cenário

sb Índice da *frame* onde ocorreu a última transição

TH Valor do limiar adaptativo

T_h Limiar superior

T_l Limiar inferior

w_r Factor de ponderação para a região

μ Média

σ Desvio padrão

ω Fator de compensação

Capítulo 1

Introdução

Hoje em dia, com o rápido desenvolvimento das novas tecnologias de informação, nomeadamente Internet e multimédia, a quantidade de informação em suporte multimédia armazenada em acervos digitais atinge valores sem precedentes. A necessidade de poder aceder e navegar nos conteúdos multimédia de forma eficiente deu origem à crescente procura por sistemas que permitam uma eficiente indexação e anotação de vídeos.

No passado a informação era recolhida na forma de unidades discretas (por exemplo numéricas e alfanuméricas) e utilizando mecanismos como Linguagem de Consulta Estruturada SQL, do inglês Structured Query Language ou, no caso da Internet, hiperligações. Em contraste, dados multimédia são volumosos e contêm uma grande quantidade de informação tornando necessária a existência de novas tecnologias de modo a que a informação relevante possa ser extraída destes meios [2].

No caso particular de conteúdo com informação visual (vídeo e imagens) esta problemática é agravada - uma vez que uma mesma imagem pode ser interpretada e percebida de forma diferente por diferentes pessoas. Deste modo, é impossível representar todas as diferentes interpretações da mesma informação visto que não é possível antever todos os cenários e contextos em que esta seria procurada e utilizada. Além disso, palavras-chave também não são capazes de representar a

natureza temporal do vídeo ou suportar relações semânticas (por exemplo: procurar numa base de dados visual (vdb, do inglês *visual database*) onde se encontra um objeto semelhante a uma determinada imagem) [2].

Este tema tem sido impulsionado pela recente tendência das empresas de *media*, e em especial aquelas responsáveis pela produção e distribuição de conteúdos noticiosos, em investir em bibliotecas digitais de vídeo, tanto para consumo interno como para disponibilização ao público. Em termos de utilização interna uma eficiente indexação agiliza todo o processo de reutilização de material de vídeo e o seu arquivo. Esta reutilização é identificada por Bertini *et al* como sendo capaz de aumentar a qualidade de produção de novos conteúdos oferecendo uma maior profundidade e contexto histórico a eventos recentes. Por outro lado a existência de arquivos de vídeo devidamente catalogados e anotados permite uma produção de conteúdos informativos mais rápida e eficiente [6]. Bertini *et al* descreve ainda o exemplo de imagens de uma qualquer cena de crime. Estas imagens poderiam ser utilizadas mais tarde para oferecer algum contexto histórico.

Apesar de tudo, se se tiver em conta a título de exemplo os programas de caráter noticioso, nem sempre é possível reutilizar o vídeo existente na sua totalidade dado que a informação contida no discurso do pivô ou no texto do oráculo poder-se tornar obsoleta após algum tempo. Deste modo, uma eficiente reutilização de imagens de vídeo só é possível se a meta informação for suficientemente completa e rica de maneira a que o sistema possa filtrar automaticamente os excertos que não são passíveis de reutilização.

Em relação aos blocos noticiosos existe uma estrutura be definida que permite delimitar os blocos nas suas peças constituintes.

Essa estrutura, e no que diz respeito à informação do vídeo, tem fronteiras de peças percetíveis, na medida em que não é frequente utilizar uma grande variedade de efeitos especiais e transições. A iluminação e o movimento da câmara são geralmente bem controlados sendo que alguns dos principais obstáculos se prendem com a presença de alguns *flashs* fotográficos [6].

A próxima secção terá como objetivo apresentar os conceitos fundamentais à compreensão dos algoritmos desenvolvidos.

1.1 Unidades estruturais de um vídeo

As unidades estruturais de um vídeo de caráter noticioso podem ir desde a simples *frame* a programas completos sendo visível uma consistência em termos de diferentes propriedades. Estas propriedades podem ser de cariz visual, estrutural ou semântico. Deste modo define-se segmento de vídeo como sendo uma porção de um vídeo onde um conjunto de propriedades é consistente durante esse mesmo segmento [7].

Na segmentação temporal de vídeo as unidades estruturais de vídeo podem ser dispostas sob a forma de uma hierarquia de níveis. A hierarquia mais consensual em toda a bibliografia consultada,encontra-se apresentada na figura 1.1.

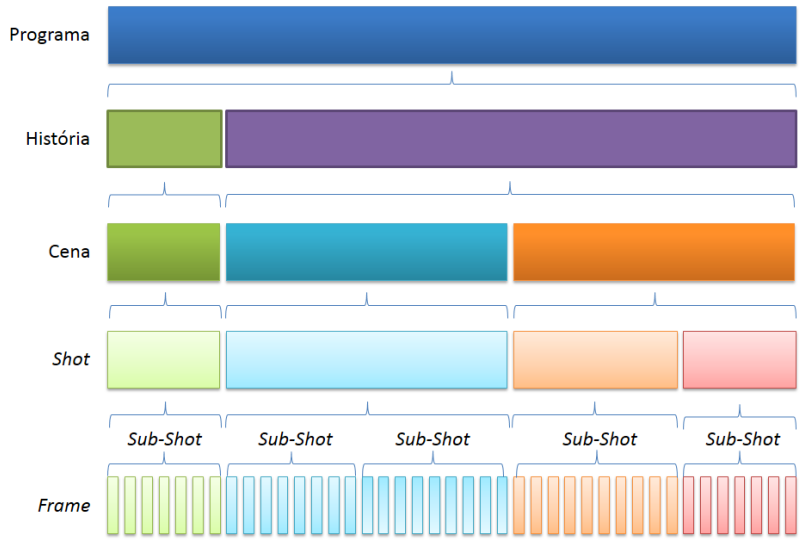


Figura 1.1: Níveis hierárquicos considerados num processo de segmentação.

Nas subsecções seguintes (1.1.1 a 1.1.5) ir-se-á detalhar cada um destes níveis hierárquicos.

1.1.1 *Frame*

No nível de hierarquia mais baixo pode-se encontrar *frames* individuais sendo que cada *frame* é uma das várias imagens estáticas que constituem um qualquer vídeo.

O termo tem a sua origem no facto de, historicamente, cada imagem ser gravada num pedaço de filme fotográfico que quando era analisado se assemelhava a uma fotografia que tinha sido "emoldurada" (do inglês *framed*). Um exemplo de uma *frame*, como seria observada num rolo de filme, pode ser encontrada na figura 1.2.

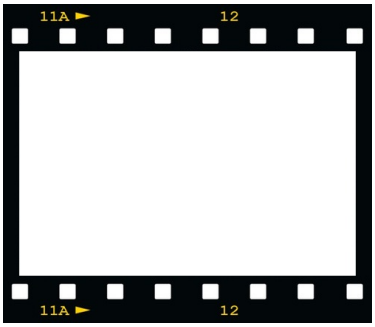


Figura 1.2: Exemplo de uma *frame* (retirado de [1]).

O número de *frames* por unidade de tempo pode ser também utilizado para descrever a duração de eventos, como transições, tendo em conta que a duração efetiva em segundos poderá variar consoante a *frame rate* de cada vídeo. Esta taxa varia a nível global. Por exemplo na América do Norte e Japão a norma para a emissão de televisão é de 30 frames por segundo (fps, do inglês *frames per second*), sendo comum a utilização de 24 fps em alta-definição. No resto do mundo, incluindo em Portugal, a norma é 25 fps.

Este número é particularmente importante no âmbito deste trabalho pois uma das metas a atingir é o processamento em tempo real de um vídeo. Deste modo qualquer algoritmo do tipo *online*, isto é, que processe vídeo em tempo real, deverá processar cada *frame* em menos de 40 ms ($1/25$ de um segundo).

Quando uma *frame* é escolhida para representar um determinado segmento, quer seja para tarefas de segmentação ou visualização, é-lhe dada a designação de *keyframe*.

1.1.2 *Sub-shot*

Petersohn introduz o conceito de *sub-shot* como sendo uma unidade acima do nível da *frame*.

Num *sub-shot* cada uma das *frames* constituintes partilham entre si determinadas características [7]. Estas características podem passar pelo movimento, luminância, crominância, etc...

Este nível hierarquico é geralmente utilizado de modo a facilitar a seleção de *keyframes* dado que os níveis superiores poderão ser demasiado heterogénios para serem resumidos por apenas uma *frame*.

1.1.3 *Shot*

No próximo nível hierárquico pode-se encontrar os *shots*.

Um *shot* consiste numa sequência contínua de *frames*, no tempo e no espaço, capturada numa uma única operação de câmara [8][7].

Geralmente é a partir deste nível que se processam as análises ao conteúdo, indexação e classificação, dos vídeos.

Neste nível hierárquico, as transições entre *shots* podem ter uma duração variável sendo comum a sua classificação consoante a sua duração em transições abruptas e transições suaves.

Transições abruptas

As transições abruptas destacam-se pelo facto de a primeira *frame* de um qualquer *shot* ser imediatamente precedida pelo último *frame* do *shot* anterior. Como exemplo de uma transição abrupta tem-se a figura 1.3.

Figura 1.3: Exemplo de uma transição abrupta entre as *frames* 3 e 4.

Transições suaves

Uma transição suave é, por sua vez, uma transição que se realiza ao longo de várias *frames*.

Transições suaves são, segundo Koprinsk *et al*, mais difíceis de detetar pois devem ser distinguidas do movimento da imagem (movimento este originado quer pela câmara, quer pelos objetos presentes em cada *frame*). Deste modo torna-se particularmente complicado detetar transições suaves, que envolvam *shots* com movimento [9].

***Fade*:** Este tipo de transição pode ser classificado em dois tipos distintos *fade-in* e *fade-out*.

Fade-in pode ser definido como o aumento (ou diminuição) gradual de intensidade de uma imagem a partir de uma *frame* de cor sólida (geralmente preta ou branca, respetivamente).

Um exemplo de um *fade-in* pode ser encontrado na figura 1.4. Neste caso a *frame* inicial é preta e a imagem aparece através do aumento de intensidade.

Figura 1.4: Exemplo de um *fade-in*.

Por outro lado um *fade-out*, tal como o nome indica pode ser definido como a alteração do nível de intensidade de uma *frame* até que esta se transforme numa cor sólida. No caso da figura 1.5, houve uma diminuição do nível de intensidade

até a imagem ficar totalmente a negro.

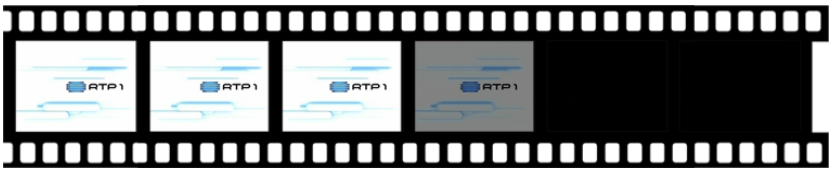


Figura 1.5: Exemplo de um *fade-out*.

Para além disso é possível que dois *fades* sejam encadeados tal como se encontra representado na figura 1.6.

Figura 1.6: Exemplo de um *fade-out* seguido de um *fade-in*.

***Dissolve*:** Este efeito acontece quando uma imagem é sobreposta a outra e vai ficando cada vez mais destacada enquanto que a outra fica cada vez mais obscurecida. Como se pode constatar pela figura 1.7.

Figura 1.7: Exemplo de um *dissolve*.

***Wipe*:** Este tipo de transição é geralmente considerado como um efeito especial e é frequentemente utilizado para suavizar a transição entre duas cenas [10]. Um exemplo deste tipo de transição pode ser observado na figura 1.8.

Esta transição ocorre quando uma fronteira móvel percorre toda a *frame* revelando o *shot* seguinte. Esta fronteira pode consistir numa única linha ou, em alguns casos, pode consistir num conjunto de linhas criando uma forma geométrica.

Figura 1.8: Exemplo de um *wipe*, nomeadamente um *zoom*.

1.1.4 Cena

A cena encontra-se no nível hierárquico imediatamente acima do *shot* e é definida como um conjunto de um ou mais *shots* adjacentes que têm em comum a mesma localização espácio-temporal. Estes são conceitos semânticos e como tal não têm uma correspondência direta com características visuais de baixo-nível. No entanto as características visuais correspondem a um conteúdo visual específico que é dependente do local e dos objetos ou pessoas representados. Deste modo dois *shots* que possuam o mesmo cenário e/ou objeto irão partilhar algumas características entre si como, por exemplo, a cor.

Sundaram *et al* descreve um modelo para as cenas passíveis de ser processadas [11][7]. Este modelo introduz o conceito de *Computable Scenes* que são, segundo os autores, um segmento que apresenta consistência no que se refere a três propriedades: cor, luminosidade e som ambiente. Estas três propriedades podem ser mapeadas em termos de características de baixo-nível.

Estes mesmos autores distinguem ainda dois tipos básicos de cena: cenas do tipo N e cenas do tipo M. As cenas do tipo N ou cenas "normais", são cenas que se enquadram perfeitamente na sua definição de *Cumputable Scenes*. Por outro lado as cenas do tipo M (de montagem ou MTV, dado que as cenas de um videoclipe musical se encontram geralmente nesta categoria) são cenas que apesar de possuírem características visuais bastante diferentes criam um tema pela maneira como são justapostas e por apresentarem uma consistência em termos de som ambiente.

1.1.5 História

Por sua vez, uma história é considerada como sendo um conjunto de cenas que partilham o mesmo tópico. Apesar disso, esta distinção entre cena e história nem sempre é feita. No entanto, para o caso específico de segmentos noticiosos esta separação é imperativa. Por exemplo, um telejornal pode passar uma reportagem sobre o aumento da criminalidade (tópico) onde poderá mostrar várias cenas em que cada uma delas é referente a um assalto individual.

1.2 Segmentação por *shots*

A segmentação de vídeos em *shots* é um processo que geralmente considerada uma de entre duas abordagens de base.

A primeira abordagem é baseada na meta-informação inserida por câmaras digitais que, para além da data e hora, já colocam muitas vezes referências geoespaciais. Estas informações são posteriormente analisadas por um algoritmo de segmentação que analisa a referência temporal de cada *frame* e, no caso de encontrar algum salto temporal, entre dois *frames* adjacentes, segmenta o vídeo com resultados ótimos. Esta informação é geralmente retirada durante o processo de edição sendo que qualquer algoritmo que segmente vídeo após a sua edição não poderá assentar exclusivamente sobre este tipo de análise.

A segunda abordagem à segmentação de vídeo consiste numa análise do conteúdo do mesmo sendo geralmente constituída pela extração de uma, ou mais, características e por um algoritmo de deteção de *shots*.

Dado que os programas noticiosos são difundidos após a edição, e tendo em conta o meio de difusão, qualquer algoritmo que se desenvolva deverá extrair características para posteriormente se proceder à deteção.

1.2.1 Extração de características

A obtenção de características relevantes à segmentação do vídeo em *shots* é feita no seu domínio espacial e temporal.

Domínio espacial

O domínio espacial refere-se ao tamanho relativo da região do vídeo de onde a característica foi calculada. Uma maior região apresenta, na maioria dos casos, um nível mais elevado de invariância enquanto que regiões mais pequenas apresentam uma maior diferenciação entre *frames* adjacentes [7].

Tendo em conta a dimensão da região, podem-se identificar três tipos de regiões, nomeadamente:

Píxel: Ao nível do píxel podem ser retiradas características como por exemplo a crominância, luminância ou contornos. Estas características são relativamente descriminantes reagindo até a pequenas alterações na imagem [7].

Região: Alternativamente podem-se extrair características ao nível da região [12] [7]. Estas características são geralmente de alto-nível pelo que não é possível calcula-las ao nivel do píxel. Uma característica possível seria, por exemplo, a presença de um determinado objeto na imagem.

Estas regiões, poderão ter diferentes tamanhos e disposições não cobrindo necessariamente toda a *frame* da mesma forma.

Petersohn afirma que para que uma abordagem seja considerada como utilizando regiões cada região deverá ter, pelo menos, 64 píxeis e deverão existir duas regiões por cada *frame* [7]. Como foi afirmado anteriormente ao aumentar o tamanho de uma determinada região estamos ao mesmo tempo a diminuir a sua sensibilidade ao ruído e movimento.

***Frame*:** Ao nível da *frame* alguns trabalhos foram publicados [13] [7].

A extração ao nível da *frame* tem como consequência a criação de características pouco sensíveis ao movimento e ao ruído. Por este motivo se dois *shots* adjacentes forem semelhantes no que diz respeito à característica escolhida poderá dar-se o caso da transição não ser detetada.

Um exemplo de uma característica calculada ao nível da *frame* é o número de objetos (e a sua localização) na mesma.

Domínio temporal

No que diz respeito ao domínio temporal todos os algoritmos de extração (baseados no conteúdo do vídeo) analisam a alteração de uma característica num determinado intervalo de tempo.

Frames adjacentes: A principal vantagem da análise de *frames* adjacentes para o cálculo da característica, e consequente segmentação, é a baixa complexidade e memória requerida por esta técnica. Por outro lado, a sensibilidade ao movimento, ao ruído, e às alterações momentâneas da característica (como *flashes* ou movimentos bruscos) poderá dar origem a vários falsos alarmes.

Janela temporal: Várias abordagens tentam resolver o problema inerente à técnica de *frames* adjacentes utilizando para isso várias *frames* para o cálculo da característica. Estas janelas temporais possuem geralmente uma dimensão que se mantém constante ao longo de todo o vídeo.

***Shot*:** Outras características incluem no seu cálculo todas as *frames* desde o início do *shot* atual. Este tipo de características considera uma janela temporal de tamanho variável onde o início da janela coincide com o início do shot.

1.2.2 Deteção de *shots*

A deteção consiste na tomada de decisão sobre a presença, ou não, de uma transição face às características produzidas préviamente.

A grande maioria dos algoritmos de deteção utiliza limiares de deteção fixos ou adaptativos. A deteção de um *shot* dá-se quando o valor da curva característica é superior ao valor do limiar num determinado ponto.

Limiar fixo

Limiares fixos são limiares que, tal como o nome sugere, não variam durante a análise de um determinado vídeo sendo determinados previamente durante a fase de treino.

Na literatura existem vários autores que recorrem a uma análise estatística da característica em zonas de transição por forma a otimizar os resultados obtidos [14]. No entanto, uma das grandes limitações da utilização deste tipo de limiares é o fato de diferentes segmentos de vídeo necessitarem de diferentes valores de limiar [7].

Limiar adaptativo

As limitações apresentadas por um limiar fixo podem ser minimizadas se este limiar for adaptável à heterogeneidade que caracteriza os diferentes segmentos de um vídeo, sendo calculado para regiões mais pequenas ou, no caso extremo, para cada *frame*.

Um limiar adaptativo parte deste principio e calcula para cada *frame* um valor de limiar específico com base nos valores anteriores da curva característica.

Kim *et al* propõe uma metodologia em que o valor de limiar considerado é a média dos valores da característica desde o último *shot* multiplicado por um fator de ponderação [15].

A existência de transições suaves, tais como *fades* e *wipes*, constituem um problema para os limiares adaptativos e fixos. Deste modo a metodologia de *twin-comparison* leva em consideração a diferença acumulada entre as *frames* numa transição gradual [2][14].

Twin-comparison

Este método requer a utilização de dois limiares fixos, um limiar superior T_h e um limiar inferior T_l.

Numa primeira fase, o limiar superior é utilizado para a deteção de transições abruptas. Em seguida, o valor da característica para as restantes *frames*, é comparado com o limiar T_l. Caso algum valor seja superior a este patamar é declarado como um potencial início de transição (F_s). Esta *frame* é subsequentemente comparada com as seguintes e a diferença entre o valor da característica e T_l é adicionada até que o valor cumulativo seja superior a T_h ou até que alguma *frame* tenha um valor inferior a T_l (como se pode verificar na figura 1.9).

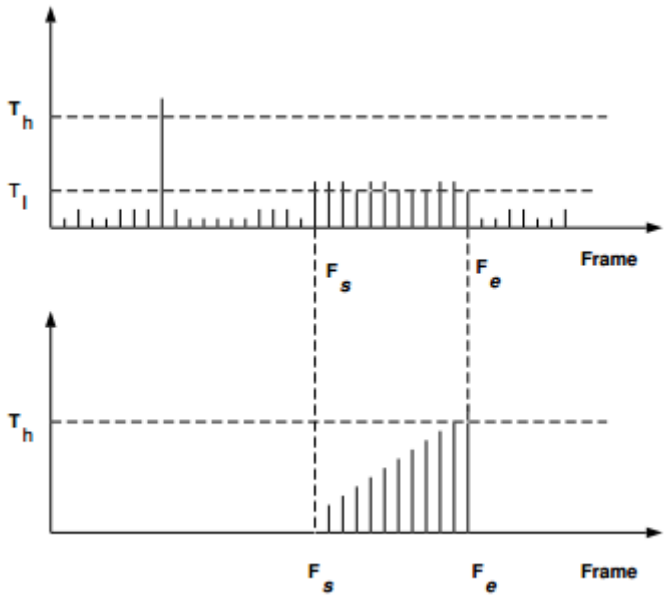


Figura 1.9: Ilustração do duplo limiar utilizado (retirado de [2]).

Caso o limiar T_h seja ultrapassado uma transição é declarada. Caso contrário a procura é recomeçada no próximo *frame* que ultrapasse T_l.

Agrupamento

As abordagens apresentadas anteriormente utilizam limiares sobre uma qualquer métrica calculada entre duas ou mais *frames*. No entanto, esta estratégia está muito dependente do tipo de vídeo considerado [9].

Para minimizar esta limitação alguns autores aplicam um algoritmo de agrupamento[1] [9].

Consideram-se geralmente dois grupos possíveis: "ocorreu mudança de *shot*"e "não ocorreu mudança de *shot*". Utilizando então um algoritmo de agrupamento, por exemplo *K-means*, cada *frame* é associada a um grupo.

A utilização deste tipo de metodologias apresenta algumas vantagens, sendo as principais: a sua versatilidade (podendo ser aplicadas a várias características e situações), a possibilidade de se poder utilizar várias características em simultâneo, o facto de dispensar a utilização de limiares e o facto de poder ser utilizada *on-the-fly*, isto é, cada *frame* é classificada à medida que é capturada [7].

Aprendizagem automática

Recentemente tem-se vindo a notar um aumento do número de abordagens que utilizam algoritmos de aprendizagem automática (tais como as máquinas de suporte vetorial, SVM, ou redes neuronais) de modo a contornar os problemas levantados pela utilização de limiares fixos [7].

Sendo geralmente rápidos de implementar, uma vez que não necessitam de uma definição explícita de qual a melhor estratégia para obtenção de limiar, podem fundir várias características e classificá-las por ordem de relevância.

No entanto, este facto aumenta o tempo de computação, visto que todo o cálculo é geralmente feito numa única fase sendo necessário ter todas as características disponíveis antes do início do processamento. Note-se ainda que estes algoritmos precisam muitas vezes de um grande conjunto de treino que seja representativo da maior parte das situações (senão de todas) que possam ocorrer.

[1] Do inglês clustering

1.2.3 Estado da arte

Os algoritmos e metodologias para a segmentação de vídeo por *shots* têm vindo a ser desenvolvidos e aplicados ao longo do tempo sendo atualmente citados em dezenas de artigos e trabalhos. Deste modo, procedeu-se então a um levantamento dos trabalhos mais referenciados sobre a segmentação digital de vídeo que será apresentado em seguida:

Diferença entre píxeis

A diferença píxel-a-píxel, ou *template matching*, consiste na comparação de duas imagens, neste caso *frames*, e pode ser formulada segundo a equação 1.1.

$$d\left(f_{i-1}, f_i\right) = \sum_{j=0}^{j<M} \sum_{k=0}^{k<N} \left|f_{i-1}\left(j,k\right) - f_i\left(j,k\right)\right| \tag{1.1}$$

Onde a *frame* f_i considerada tem a dimensão M por N.

Esta técnica é bastante sensível a ruído e movimento quer dos objetos quer da câmara visto que baseia-se na posição absoluta de cada um dos píxeis. Jiang *et al*, refere que este método é, portanto, bastante suscetível a falsos alarmes. Este mesmo autor afirma ainda que este efeito pode ser minimizado através do agrupamento dos píxeis de cada *frame* em regiões ou através de um simples filtro 3×3 [16] [14].

Outra versão deste mesmo algoritmo consiste em contar o número de píxeis cujo valor é significativamente alterado (valor este que é obtido através da comparação do nível de intensidade de um determinado píxel com um primeiro limiar). Caso o número de píxeis ultrapasse um segundo limiar o algoritmo conclui então que está na presença de uma transição.

Histograma

Os métodos baseados na análise do histograma são os mais comuns visto que a cor não sofre grandes alterações dentro do *shot*. Estes métodos são também utilizados pois são bastante resistentes a ruído e ao movimento.

Distância euclidiana: Danisman *et al*, propôs um algoritmo que calcula o histograma, no espaço RGB, e a distância Euclidiana entre cada *bin* de duas *frames* sucessivas (como se pode observar pela equação 1.2). A transição ocorre quando a distância calculada é superior a um determinado limiar [17].

$$d\left(f_{i-1}, f_i\right) = \sqrt{\sum_{j=0}^{j<B} \sum_{k=0}^{k<3} \left[H_{i-1}\left(j,k\right) - H_i\left(j,k\right)\right]^2} \tag{1.2}$$

Sendo B o número de *Bins* em que cada uma das cores k se decompõe e H a função de histograma.

Neste trabalho são utilizados três limiares: um para a deteção de cortes abruptos, outro para transições suaves e um terceiro para diminuir a redundância temporal presente em cada *frame*. Afirmam que *frames* consecutivas têm informação espacial semelhante pelo que não é necessário processar todas as *frames*. Este terceiro limiar reduz, portanto, o tempo de computação necessário à execução do algoritmo.

***Chi-squared*:** A métrica *Chi-squared* (χ^2) é calculada através da distância entre duas *frames* (f_i e f_{i-1}) segundo a equação 1.3.

$$d\left(f_{i-1}, f_i\right) = \sqrt{\sum_{j=0}^{j<B} \sum_{k=0}^{k<3} \left[H_{i-1}\left(j,k\right) - H_i\left(j,k\right)\right]} \tag{1.3}$$

Segundo Jiang *et al* vários investigadores afirmam que χ^2 produz melhores resultados quando comparados com outros algoritmos.

Interceção de histogramas: Interceção de histogramas é, ao contrário dos outros algoritmos aqui estudados, uma medida de semelhança, (s). Isto é, assume o seu máximo valor quando ambas as *frames* são iguais [18]. O algoritmo pode ser definido como a soma do mínimo valor entre cada *bin* (como se pode observar na equação 1.4).

$$s\left(f_{i-1}, f_i\right) = \sum_{j=0}^{j<B} \sum_{k=0}^{k<3} min\left(H_{i-1}\left(j,k\right); H_i\left(j,k\right)\right) \tag{1.4}$$

Esta técnica, combinada com algoritmos de compensação de movimento produz bons resultados como demonstra Fang *et al* [19].

Contornos

Relativamente à deteção utilizando os contornos de uma *frame*, Zabih *et al* desenvolveu um método onde o detetor de contornos Canny foi aplicado a cada *frame* e após o alinhamento destes era possível calcular a percentagem de contornos comuns a ambas as *frames*. Em termos da deteção de picos utilizou-se como métrica a distância de Hausdorff. Este método mostrou-se bastante robusto no que diz respeito ao ruído e ao movimento de câmara [20].

Estatísticas MPEG

Lelescu and Schonfeld apresentam uma abordagem estatística que trabalha sobre vídeos MPEG comprimidos. Após a extração dos valores médios de luminância e crominância para cada bloco é feita uma análise *Principal Component Analysis* (PCA) ao vetor resultante [21] [22].

Este processo foi efetuado apenas para as N primeiras *frames* de cada *shot*.

A característica resultante é modelada segundo uma distribuição Gausseana em que a média e cóvariancia são estimadas utilizando as N primeiras *frames*.

Para cada nova *frame* a característica é estimada com base na metodologia da semelhança máxima (*Generalized Likelihood Ratio*) sendo detetado uma nova transição quando este valor ultrapassava um limiar pré-definido.

Outras

Redes Neuronais: A técnica desenvolvida por Lineheart deteta apenas *dissolves* através de um classificador treinado (rede neuronal), recorrendo a histogramas no

espaço de cor YUV, gradientes direcionais, e contraste baseado em contornos. O classificador deteta a existência de um possível *dissolve* em várias escalas temporais e agrupa os resultados utilizando uma estratégia *winner-take all* [21].

O classificador é treinado utilizando um sintetizador de *dissolves* que cria *dissolves* artificiais a partir de uma qualquer fonte de vídeo [21].

Foveated: Segundo Bocciognone *et al* os seres humanos conseguem detetar mudança de *shots* movendo os olhos três ou quatro vezes em cada segundo e fixando alguns pontos de interesse (Figura 1.10).

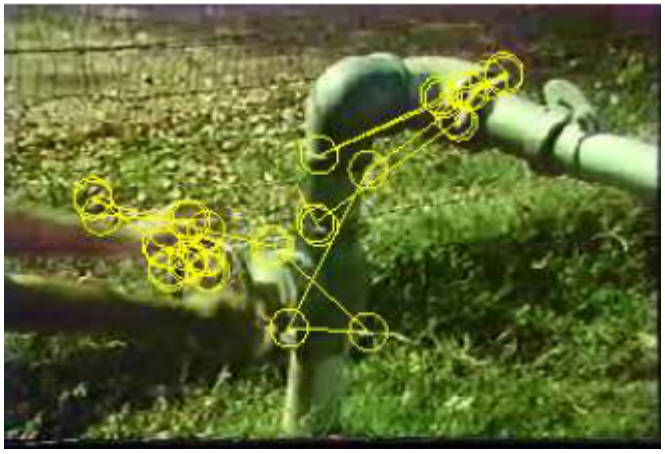


Figura 1.10: Movimento dos olhos de um observador (retirado de [3]).

Partindo deste princípio, o autor desenvolveu um algoritmo que em cada instante calcula uma medida de consistência entre os pontos de interesse presentes em cada *frame* [3].

Os algoritmos e metodologias aqui apresentados pretendem ser um resumo do que mais relevante se faz na segmentação de vídeo em *shots*. Na proxima secção irá-se apresentar os principais trabalhos relativamente à segmentação do vídeo por histórias.

1.3 Segmentação por história

Acima do nível do *shot* aparece a segmentação em cenas e histórias. No caso dos programas noticiosos uma mudança de história nem sempre se encontra associada a uma mudança de cena. Pelo que para efeitos de metodologia de segmentação por história irá considerar-se que a história é o nível imediatamente acima do *shot*, sendo a sua segmentação efetuada recorrendo às *keyframes* selecionadas no processo de segmentação por *shots*.

Segundo o National Institute of Standards and Technologies (NIST) uma história pode ser definida como um segmento de um programa noticioso que possui um foco coerente que poderá incluir assuntos politicos, relatórios financeiros, meteorologia, segmentos desportivos, entre outros. Por outro lado segmentos não noticiosos são considerados como *miscellaneous stories* onde estão incluidos intervalos comerciais, *lead-ins*,*lead-outs*, etc [23].

1.3.1 Estado da arte

Zhai *et al*, propõe que uma abordagem à problemática da segmentação por história que assenta num *Shot Connectivity Graph* (SCG). Partindo do pressuposto que antes do início de cada história o pivô faz geralmente uma breve introdução da história, antes da reportagem de campo, é possível notar que um padrão emerge ao longo de todo o programa noticioso (figura 1.11) [23].

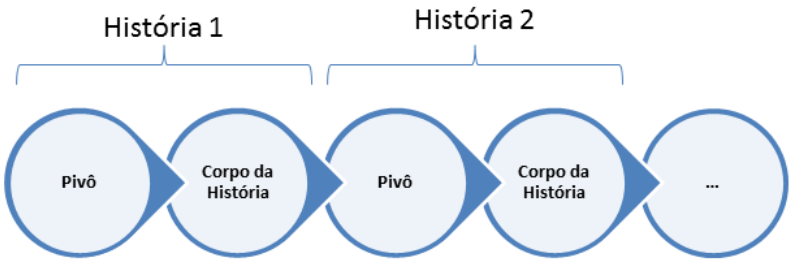


Figura 1.11: Padrão observavel em segmentos noticiosos.

Com base neste facto Zhai *et al* identifica as *keyframes* onde se encontram os pivôs e, utilizando uma métrica de semelhança entre estas, agrupa-as de forma a

formar um SCG e analisa-o de modo a verificar a presença de ciclos. Enquanto que os pivôs principais correspondem a grandes ciclos que estão presentes ao longo de todo o programa, outros segmentos (como entrevistas) irão corresponder a pequenos ciclos localizados numa área em concreto [23].

De forma muito semelhante Goyal *et al* propõe também que se utilize o padrão representado na figura 1.11 de modo a segmentar segmentos noticiosos em histórias. Goyal assenta a sua abordagem numa heurística que se baseia no facto de que um pivô irá aparecer pela primeira vez entre os 25 e os 55 segundos de um qualquer programa noticioso. Esta heurística permite encontrar uma *template* para o pivô que será usada na permitir a sua posterior identificação através de um classificador treinado [24].

Mais tarde, Misra *et al*, adiciona ao trabalho de Goyal uma vertente semântica no sentido em que calcula a semelhança semântica entre frases transcritas do vídeos utilizando uma técnica baseada na *Latent Dirichelet Allocation* (LDA) [25].

Uma abordagem alternativa aos algoritmos descritos é porposta por Xie *et al*. No seu trabalho Xie utiliza as transcrições automáticas da fala presente no vídeo, obtidas através de um sistema de reconhecimento de fala. Cada frase sé mapeada no espaço Euclideano sendo calculada uma medida de semelhança entre frases. Este processo, segundo Xie, iria tornar evidente um mapa de relações entre frases que será então analisado utilizando *Laplacian Eignmaps* [26].

Haumptman e Witbrock utilizam também transcrições automáticas do segmento noticioso de modo a proceder à segmentação por história. No seu trabalho começam por examinar a transcrição procurando por pausas longas (acima de 15 segundos) que indicariam intervalos comerciais. Esta análise de pausas é utilizada também para refinar a segmentação utilizando um limiar mais baixo (cerca de 7 segundos) para delimitar histórias individuais [27].

Poulisse *et al*, descreve uma abordagem baseada no conteúdo presente no audio, vídeo e texto do programa noticioso. Este autor utiliza, aliádas a um classificador treinado, características como a repetição de vocabulário, semelhança semântica (tambem utilizando a LDA), a estrutura típica de um programa noticioso (identificando pontos onde tipicamente ocorre uma mudança de história), o tamanho típico

de uma história, pausas no discurso, correntes lexicais e presença de palavras chave que geralmente indicam tipicamente o inicio de uma reportagem ("boa noite", "de seguida", "depois do intervalo", entre outros) [28].

1.4 Objetivos

A pesquisa bibliográfica que foi realizada permitiu constatar que existe uma lacuna no que toca à segmentação digital de vídeo em tempo real, nomeadamente no que diz respeito à segmentação por histórias. Isto porque os algoritmos mais utilizados recorrem a técnicas de agrupamento e/ou análise de ciclos. Como tal, é geralmente necessário esperar pelo final do programa noticioso antes de ser possível a tomada de decisão.

Este trabalho tem como objetivo principal desenvolver um conjunto de algoritmos que sejam capazes de segmentar um programa noticioso ao nível do *shot* e da história em tempo real.

Entende-se como processamento em tempo real, a disponibilização dos resultados da segmentação à medida que o vídeo é capturado ou lido.

Como objetivo secundário pretende-se que esta tese forneça uma base sólida para futuros desenvolvimentos nesta área, utilizando metodologias relativamente simples e flexíveis.

1.5 Solução proposta

De forma a atingir os objetivos anteriormente definidos adotou-se a implementação de um algoritmo de segmentação por *shots* e um algoritmo de segmentação por histórias segundo o diagrama da figura 1.12.

Durante o decorrer do programa, cada *frame* é analisada com vista à deteção de *shots*. A determinação destes *shots* permitirá selecionar as imagens mais representativas (*keyframes*) onde serão detetadas faces, que permitirão, num nível superior, facilitar a segmentação em histórias.

Figura 1.12: Diagrama de blocos do trabalho.

Como tal, enquanto que para o algoritmo de deteção de *shots* é necessário processar todas as *frames*, tanto a seleção de *keyframes* como a segmentação por história apenas serão executadas uma vez por *shot*. Tendo isto em conta, na figura 1.13 pode-se observar o fluxograma geral da metodologia adotada.

Figura 1.13: Fluxograma geral da solução proposta.

Note-se que esta arquitetura terá implicações diretas sobre a construção dos algoritmos. Isto porque apenas o algoritmo de segmentação por *shots* será executado para todas as *frames* e como tal terá de ser mais eficiente em termos de tempo de computação. Por outro lado o algoritmo de segmentação por histórias poderá utilizar algoritmos mais sufisticados, e como tal mais lentos, dado que não é executado tão frequentemente.

1.6 Estrutura do trabalho

No primeiro capítulo deste trabalho abordou-se a questão da segmentação digital de vídeo.

No capítulo seguinte (capítulo 2) será descrito o *corpus* que foi recolhido e anotado manualmente, com o objetivo de ser utilizados nos algoritmos de segmentação automática.

No capítulo 3 descreve-se com maior detalhe, o algoritmo de segmentação por *shots* assim como a metodologia de seleção de *keyframes* que será utilizada no processo de segmentação por histórias (capítulo 4).

A descrição das experiências que foram realizadas para avaliar os algoritmos de segmentação automática por *shots* e por história é feita no capítulo 5 oonde também será apresentada uma crítica aos resultados obtidos.

No último capítulo (capítulo 6) podem-se encontrar agumas conclusões e propostas para o desenvolvimento futuro dos algoritmos apresentados.

Capítulo 2

Corpora e anotação manual

Neste capítulo descreve-se o processo de recolha e anotação manual da *corpora* que precedeu o desenvolvimento dos algoritmos de segmentação de *shots* e histórias.

Este passo é importante de modo a conhecer a fundo a composição tipica dos programas noticiosos portugueses, nomeadamente programas da Rádio e Televisão de Portugal (RTP) e da Sociedade Independente de Comunicação (SIC), que são transmitidos no horário nobre. O conjunto dos vídeos recolhidos no âmbito deste trabalho foi dividido em dois sub-conjuntos denominados *corpus* de teste e *corpus* de treino.

Estes dois sub-conjuntos serão segmentados e anotados segundo utilizando a ferramenta Anvil, secção 2.1. A composição de ambos os conjuntos será analisada na secção 2.2.

2.1 Anotação do *corpus*

O *corpus* foi recolhido, na sua grande maioria, em Agosto de 2012 utilizando um disco multimédia que grava os vídeos com uma resolução de 720×576 em MPEG. Estes vídeos foram posteriormente convertidos, utilizando uma aplicação chamada VirtualDub, para MP42 (MPEG 4, versão 2) em *quarter video graphics array* (QVGA, ou seja 320×240 píxeis de resolução) a 25 fps.

No que se refere à segmentação manual vários cuidados foram tomados de modo a que a avaliação do algoritmo fosse a mais detalhada possível, nomeadamente: Para além da localização no vídeo das transições que indicam a mudança de *shot* também é importante registar a duração das mesmas. Isto porque algumas transições podem durar dezenas de *frames* e a deteção automática pode ocorrer em qualquer uma destas. Foi registado o inicio e o fim do intervalo do segmento noticioso de modo a que as transições detetadas neste espaço não sejam consideradas no processo de avaliação. No caso dos *shots* foi ainda anotado o tipo de transição ocorrida.

Para realizar a segmentação e indexação manual do *corpus* utilizou-se uma ferramenta de anotação de vídeo chamada Anvil [29]. O programa Anvil, além de ser gratuito, possui uma interface com o utilizador relativamente simples permitindo uma anotação multicamada e a exportação para um documento *Extensible Markup Language* (XML) que possui a informação da anotação.

No que toca à interface com o utilizador a aplicação pode ser dividida em quatro componentes principais (figura 2.1):

No primeiro painel podemos carregar o vídeo, ou ficheiro de anotação que desejamos editar. Este painel possui ainda comandos para a edição do ficheiro de especificação ou para proceder a análises sobre o ficheiro de anotação.

No segundo e terceiro painel poderemos encontrar o vídeo e informações especificas sobre a *frame* seleccionada na *timeline*, respetivamente.

Por último podemos encontrar a *timeline* onde é efetuada a anotação propriamente dita.

O documento gerado pelo Anvil que inclui a informação de anotação manual

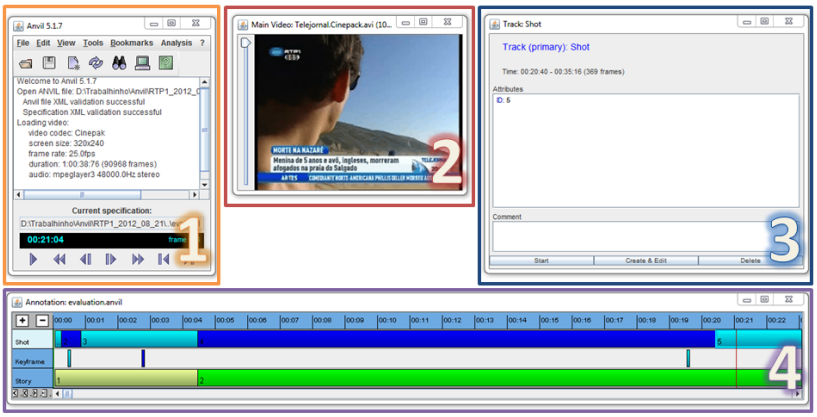


Figura 2.1: Ambiente de trabalho do *software* de anotação manual de vídeo Anvil.

respeita uma estrutura que é definida por um ficheiro de especificação, também no formato XML.

A anotação manual do programa noticioso considerou três grupos principais: *Shots*, Cenas e ProgramUnits.

No primeiro grupo colocou-se informação relativa à segmentação do programa por *shots*. Atríbuiu-se um identificador para cada *shot* e caracterizou-se a duração e o tipo de transição entre *shots*.

Do mesmo modo, no grupo das cenas, foram consideradas duas camadas principais: Numa camada encontra-se um identificador único para cada Cena e noutra um descritivo com a localização e tipo de transição.

No terceiro grupo poderemos encontrar por sua vez informação relativa ao conteúdo do vídeo. Isto é, informação relativa ao inicio e fim de histórias, jingles, presença do pivô, publicidade ou *filler*.

2.2 Estatisticas do *corpus*

De modo a desenvolver, treinar e avaliar os algoritmos de segmentação foi necessário dividir o *corpus* recolhido em dois conjuntos, conjunto de treino e conjunto de teste.

2.2.1 *Corpus* de treino

De modo a treinar, e afinar, alguns aspetos dos algoritmos apresentados foi necessário recolher um conjunto de vídeos, intitulados como *corpus* de treino, que foram afetos exclusivamente a esta função.

Na recolha do conjunto de treino foram considerados três programas noticiosos que passam na televisão portuguesa no horário nobre. A informação detalhada sobre cada um deles poderá ser encontrada na tabela 2.1.

Tabela 2.1: Descrição do conjunto de treino

Nome do Programa	Canal	Data da captura	Duração	Designação
Telejornal	RTP 1	18-01-2008	01:01:14	TJ_{tr1}
Telejornal	RTP 1	21-08-2012	01:00:38	TJ_{tr2}
Telejornal	RTP 1	22-08-2012	01:00:02	TJ_{tr3}

Deste modo o *corpus* de treino possui cerca de três horas de vídeo.

Da anotação manual dos programas por *shots* foi elaborada uma estatistica associada aos diversos tipos de transição. Esta estatística encontra-se na tabela 2.2.

Tabela 2.2: Resultado da segmentação manual de *shots* do conjunto de treino

Elemento do *corpus*	Transições	Fades [%]	Wipes [%]	Dissolves [%]	Cuts [%]
TJ_{tr1}	459	0.7	2.4	15.9	81.0
TJ_{tr2}	535	0.2	2.8	96.4	0.6
TJ_{tr3}	611	0.8	1.1	16.0	82.0
Total	1605	0.6	2.1	42.8	54.6

Como se pode verificar pela tabela 2.2 o *corpus* de treino possui um total de 1605 *shots* sendo que 45.5% são transições suaves e 54,6% são trasições abruptas.

A publicidade correspondendo a 13 minutos e 26 segundos das três horas de *corpus* (7,4%) não foi considerada na deteção de eventos. Isto porque blocos

publicitários possuem uma estrutura completamente diferente do restante programa e como tal a metodologia de segmentação não poderia ser a mesma.

No que toca à segmentação por histórias o *corpus* de treino possui um total de 84 segmentos.

2.2.2 *Corpus* de teste

À parte do grupo de treino procedeu-se à segmentação de um conjunto adicional de programas noticiosos. Estes programas constituem o conjunto de teste (CTe). Na tabela 2.3 podem-se encontrar informações relativas a estes vídeos.

Tabela 2.3: Descrição do conjunto de teste

Nome do Programa	Canal	Data da captura	Duração	Designação
Telejornal	RTP 1	12-01-2012	01:03:49	TJ_{ts1}
Telejornal	RTP 1	23-08-2012	01:01:42	TJ_{ts3}
Telejornal	RTP 1	24-08-2012	01:02:29	TJ_{ts3}
Jornal da Noite	SIC	24-08-2012	00:53:32	JN_{ts4}

Deste modo o *corpus* de teste possui cerca de quatro horas de vídeo. Estes vídeos possuem um total de 2277 *shots* sendo que 67.8 % correspondem a transições suaves e 32,2% a transições abruptas (tabela 2.4).

Tabela 2.4: Resultado da segmentação manual de shots do conjunto de teste

Elemento do *corpus*	Transições	Fades [%]	Wipes [%]	Dissolves [%]	Cuts [%]
TJ_{ts1}	643	1.7	3.1	89.0	6.2
TJ_{ts2}	546	0.7	1.8	81.0	16.5
TJ_{ts3}	607	0.2	1.3	63.3	35.3
JN_{ts4}	481	1.9	3.3	49.1	45.7
Total	2277	1.1	2.4	64.3	32.3

No *corpus* de teste a publicidade corresponde a 15 minutos (6,4%) e, tal como aconteceu com o conjunto de treino, não serão anotados eventos nestas zonas do

vídeo.

Com os dados recolhidos nas tabelas anteriores é possível verificar que a grande maioria das transições presentes num programa noticioso típico são *dissolves* ou *cuts*.

No entanto, na *corpora* recolhida as transições abruptas têm uma maior percentagem no conjunto de treino(54.6%) que no conjunto de teste (32.3%). Este fato é justificado pelo aumento proporcional do número de *dissolves* detetados (64.3% no conjunto de teste e 42.8% no conjunto de treino).

2.3 Conclusão

Neste capítulo decreveu-se a metodologia de anotação manual do *corpus* de de treino e de teste assim como a sua constituição.

Para a tarefa de segmentação e anotação manual utilizou-se a ferramenta de anotação de vídeo Anvil. Desta forma é possível exportar a informação de segmentação manual para um ficheiro XML que será lido pelo script de avaliação.

Capítulo 3

Algoritmo de segmentação por *shots*

Neste capítulo será apresentado o algoritmo para a segmentação temporal de *shots*.

Este algoritmo baseia-se no trabalho proposto por Zabih *et al* e no limiar adaptativo desenvolvido por Kim *et al* [20] [15]. Os diferentes passos deste algoritmo podem ser observados na figura 3.1 e serão executados para cada uma das frames do vídeo.

O algoritmo de segmentação por *shots* utiliza como característica os contornos da imagem de cada *frame* (secção 3.1). O valor desta característica será posteriormente comparado com o valor de um limiar adaptativo de forma a detetar a presença de transições (secção 3.2). Após a deteção de um novo *shot* será selecionada uma *keyframe* representativa do mesmo. O critério utilizado para o efeito será definido na secção 3.3.

A otimização do algoritmo será efetuada através de algumas experiências sobre o corpus de treino na secção 3.4.

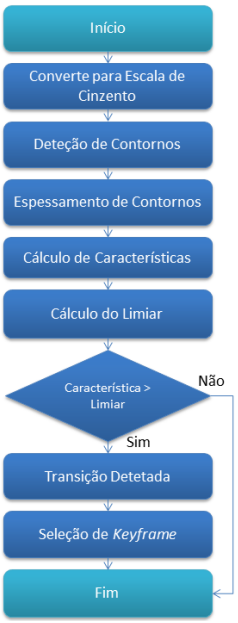


Figura 3.1: Fluxograma do algoritmo de segmentação por *shots*.

3.1 Característica de contornos

De modo a extrair a característica de contornos a imagem é adquirida utilizando uma *toolbox* de aquisição de imagem, incluída na plataforma de desenvolvimento Matlab 2012a. Através das funções desta *toolbox*, é possível extrair e processar cada *frame* a partir de um ficheiro de vídeo ou diretamente de uma placa de captura de televisão. Após a extração da *frame* RGB de 24 bits é então convertida numa imagem em escala de cinzento com 256 níveis.

Após esta conversão é feita a deteção de contornos recorrendo ao detetor de Sobel.

Este detetor (cujas matrizes estão representados em 3.1 e 3.2) foi desenvolvido por Irwin Sobel e consiste num operador que calcula uma aproximação do gradiente da intensidade dos píxeis da imagem. Em cada ponto da imagem, o resultado da aplicação do filtro Sobel devolve o gradiente ou a norma deste vetor. Distingue-se do *kernel* de Prewitt por dar uma maior importância aos *pixeís* centrais.

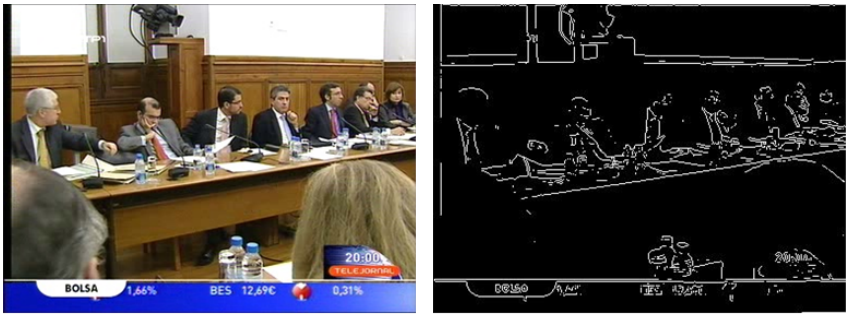
$$\frac{\partial}{\partial x} \begin{cases} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{cases} \tag{3.1}$$

$$\frac{\partial}{\partial y} \begin{cases} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{cases} \tag{3.2}$$

Tal como os operadores de Prewitt e Roberts, a deteção é efetuada através da convolução horizontal e vertical das matrizes respetivas.

Os contornos são selecionados nos pontos onde o gradiente da imagem resultante é maior, de modo a filtrar pequenos detalhes e ruído.

Após o cálculo dos contornos é feita a binarização dos mesmos (figura 3.2).



(a) Imagem adquirida (b) Imagem após deteção de contornos

Figura 3.2: *Frame* após a deteção de contornos.

Seguidamente, os contornos serão espessados através da correlação cruzada com um *template* de 25×25, tal como é possível observar na figura 3.3.

Note que a correlação cruzada de duas matrizes A (com a dimensão $M_a \times N_a$) e B (com a dimensão $M_b \times N_b$) é dada, para o par de píxeis j e k, pela equação 3.3 .

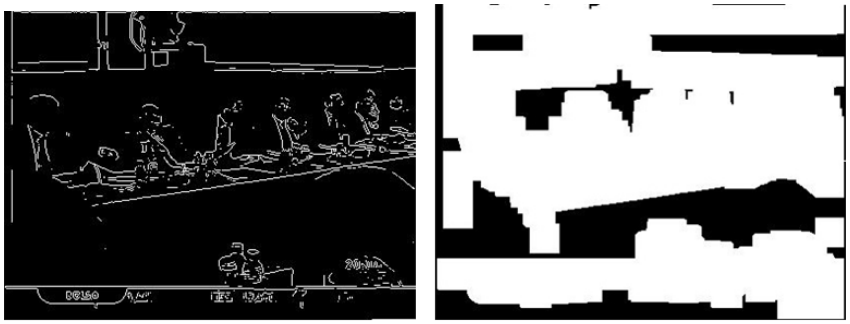
$$C(j,k) = \sum_{m=0}^{M_a-1} \sum_{n=0}^{N_a-1} A(m,n) \cdot conj(B(m+j, n+k)) \tag{3.3}$$

Onde $0 \leq j \leq M_a + M_b - 1$ e $0 \leq k \leq N_a + N_b - 1$.

No caso do algoritmo apresentado neste trabalho temos: $M_a = M$, $N_a = N$, (o que corresponde respetivamente à largura e altura da imagem) e $M_b = N_b = 25$.

A dimensão deste *template* de 25×25 foi determinada empiricamente através de vários testes realizados sobre o *corpus* de treino que serão descritos na secção 3.4.

O espessamento de contornos torna o algoritmo menos sensível ao movimento de câmara e de objetos uma vez que apenas uma fração dos contornos será alterado entre a *frame* atual e a *frame* anterior.



(a) Imagem após deteção de contornos (b) Imagem após a aplicação da *template*

Figura 3.3: *Frame* após a aplicação de uma *template* 25×25.

O valor da característica de contornos para a *frame* i, CD_i, é a soma, para as diferentes regiões, da diferença absoluta entre duas imagens de contornos calculada entre *frames* adjacentes, C_i^r e $C_{(i-1)}^r$, normalizada utilizando a área da região correspondente A_r. A equação utilizada pode ser observada em 3.4.

$$CD_i = \sum_{r=1}^{R} w_r \cdot p_r \cdot \frac{\sum \left| C_i^r - C_{(i-1)}^r \right|}{A_r} \tag{3.4}$$

Onde r é a região da imagem, R o número de regiões, p_r a fração da imagem definida pela região r, C_i^r o contorno presente na região, A_r a área da região definida em píxeis e w_r um fator de ponderação que reflete a importância relativa das diversas regiões da imagem.

Após o cálculo da característica de contornos é agora necessário proceder à deteção das fronteiras do *shot*. Para o efeito utilizou-se um limiar adaptativo que será descrito na secção seguinte.

3.2 Limiar adaptativo

Para a deteção de transições utilizou-se um limiar adaptativo que segue a tendencia do valor da característica de contornos.

O limiar adaptativo TH_i utiliza então o valor cumulativo da característica de contornos de todas as *frames* desde a última transição e é definido segundo as equações 3.5 e 3.6.

$$TH_i = \frac{\sum_{j=sb+1}^{i-1} CD_j}{i - sb + 1} \cdot \omega_i \tag{3.5}$$

Onde sb é o índice da *frame* onde ocorreu a última transição.

Como se pode observar na equação 3.5 o valor de TH_i consiste essencialmente num produto entre a média algébrica dos valores de CD, desde o inicio do *shot* até à *frame* atual, e o fator de compensação ω_i, definido em 3.6.

$$\omega_i = \frac{2 \cdot CD_{max}}{CD_i} \tag{3.6}$$

Note que ω_i corresponde a duas vezes o valor máximo de CD (CD_{max}) a dividir por CD_i.

O valor de CD_{max} acontece quando as imagens de contornos (C_i^r e C_{i-1}^r) não possuem qualquer píxel em comum, sendo a soma da diferença absoluta entre estas igual a A_r. Substituindo esta relação na equação 3.4 obtém-se a igualdade presente na equação 3.7.

$$CD_{max} = \sum_{r=1}^{R} w_r \cdot p_r \tag{3.7}$$

Deste modo, ω_i é definido segundo a equação 3.8:

$$\omega_i = \frac{2 \cdot \sum_{r=1}^{R} w_r \cdot p_r}{CD_i} \tag{3.8}$$

Este limiar possui uma vantagem clara quando comparado com os limiares adaptativos tradicionais. Isto porque um limiar adaptativo normal aproxima-se de zero quando a média de CD é baixa, o que torna bastante sensível a pequenas variações do CD_i.

Este limiar caracteriza-se por ser elevado quando o valor de CD_i for relativamente baixo, quando comparado com CD_{max}. Por outro lado quando CD_i aproxima-se de CD_{max} o parâmetro ω_i irá ser igual a 2, o que fará diminuir o valor de TH_i.

Um novo *shot* será iniciado quando CD_i for maior que TH_i.

Após ter sido detetada uma nova transição o procedimento seguinte visa identificar qual a *frame* mais representativa do *shot* (denominada *keyframe*).

3.3 Seleção de *keyframes*

É frequente representar um segmento de vídeo através de uma, ou mais, imagens desse segmento. Este processo, conhecido como seleção de *keyframes*, deve ser capaz de escolher um subconjunto das *frames* que sendo as mais representativas desse segmento não comprometem a informação que se pretende extrair posteriormente. Deste modo, uma eficiente representação do vídeo é uma parte essencial da sua análise e gestão, sumarizando os acontecimentos do mesmo e permitindo o seu processamento de forma eficiente do ponto de vista computacional [30].

Para a seleção de *keyframes* três abordagens base são geralmente utilizadas na literatura:

Numa representação ao nível da *frame* todas as *frames* do vídeo são consideradas no processamento das características, resultando num enorme volume de informação a ser processada. Esta abordagem é a que requer mais recursos de sistema apesar

de não haver qualquer perda de informação.

Noutras implementações, o vídeo é amostrado de maneira uniforme sendo a quantidade de informação disponível dependente da frequência de amostragem pretendida. Esta abordagem pode não representar o conteúdo do vídeo corretamente dado que alguns *shots* podem ser demasiado pequenos para serem representados nesta técnica.

Deste modo uma das metodologias mais populares, consiste na segmentação em *shots* onde pelo menos uma das *frames* que os constituem é selecionada como sendo uma *keyframe* [31]. As implementações mais comuns desta metodologia consistem em selecionar a *frame* central do *shot* ou uma das *frames* que o delimitam (primeira ou última).

No entanto para o processo de segmentação de histórias o principal objetivo é a deteção e identificação de pivôs. Como tal uma *keyframe* que represente um momento com muito movimento ou uma qualquer transição seria prejudicial para a *performance* do algoritmo.

Por este motivo decidiu-se que as melhores *keyframes* para o efeito seriam aquelas com menos movimento. Estando esta característica relacionada com a diferença entre *frames* podemos concluir que uma zona com pouco ou nenhum movimento possui um valor de CD_i relativamente baixo.

A deteção de pivôs será também uma operação com um elevado custo em termos computacionais pelo que considera-se que apenas uma *keyframe* por *shot* é suficiente para o processo de segmentação.

Deste modo as *keyframes* serão as *frames* cujo o valor de CD_i é menor em cada *shot*.

A seleção de *keyframes* conclui o algoritmo de segmentação por *shots* sendo este reiniciado após a análise da *keyframe* seleccionada por parte do algoritmo de segmentação por histórias.

Por forma a concluir o desenvolvimento do algoritmo de segmentação é ainda necessário proceder a algumas experiências por forma a verificar se todos os componentes do algoritmo, nomeadamente *templates*, regiões e *subsampling* estão bem

dimensionados.

3.4 Otimização e treino do algoritmo de segmentação por *shots*

De modo a obter o melhor resultado possível nos ensaios sobre o *corpus* de teste, foi necessário otimizar e testar os principais componentes do algoritmo de segmentação de *shots*. Estes componentes são: a forma e tamanho da *template* responsável pela dilatação dos contornos, a utilização de estratégias de *subsampling* e a importância de certas regiões da *frame* para o processo de segmentação.

Os ensaios, realizados exclusivamente sobre o *corpus* de treino, ajudaram a definir quais os melhores parâmetros para a segmentação do *corpus* de teste.

Por forma a ser possível comparar os resultados das várias experiências quantificaram-se os resultados em termos de verdadeiros positivos (Tp), falsos positivos (Fp) e falsos negativos (Fn). Estes dados serão utilizados para o cálculo da *Precision* (Pre) e da *Recall* (Rec) que servirão de base à métrica F_3. Esta métrica consiste numa média harmónica entre a *Precision* e *Recall* onde a *Recall* é três vezes mais importante que a precisão do algoritmo.[1]

O primeiro parâmetro a ser optimizado prende-se com a forma e tamanho da *template* utilizada durante a dilatação de contornos.

3.4.1 Definição da *template*

Um dos parâmetros susceptiveis de otimização no algoritmo de segmentação por *shots* é a área e a forma da *template* responsável pelo espessamento de contornos.

Na figura 3.4 podem-se observar as *templates* testadas nesta experiência. Note que no contexto das *templates* circulares r corresponde ao raio da *template* em píxeis.

[1] Estas métricas serão apresentadas em maior detalhe no Capítulo 5. A razão pela qual se utilizam neste capítulo prende-se pela necessidade de otimizar alguns parâmetros os algoritmo antes do início dos testes finais.

(a) Ball $r = 10$ (b) Ball $r = 15$ (c) Ball $r = 20$ (d) Ball $r = 25$

(e) Square 15×15 (f) Square 20×20 (g) Square 25×25 (h) Square 30×30

Figura 3.4: *Templates* utilizadas.

Utilizando cada uma destas *templates* obtiveram-se os resultados presentes na tabela 3.1.

Tabela 3.1: Resultados do algoritmo de segmentação por *shots* variando o *template* de espessamento.

Tipo	Tamanho	Imagem	T_p	F_n	F_p	Pre [%]	Rec [%]	F_3 [%]
Ball	$r = 10$	3.4a	1058	547	48	95.7	65.9	68.0
	$r = 15$	3.4b	1079	526	47	95.8	67.2	69.3
	$r = 20$	3.4c	934	671	30	96.9	58.2	60.6
	$r = 25$	3.4d	857	748	31	96.5	53.4	55.9
Square	15×15	3.4e	1002	603	40	96.2	62.4	64.7
	20×20	3.4f	1083	522	50	95.6	67.5	69.5
	25×25	3.4g	1088	517	47	95.9	67.8	69.8
	30×30	3.4h	1066	539	46	95.9	66.4	68.5

Como se pode observar o valor de F_3 máximo para as *templates* circulares foi encontrado com um raio igual a 15 píxeis. Por outro lado, a forma quadrada obteve o seu melhor resultado com uma *template* 25×25.

Dado que a *template* quadrada 25×25 obteve o maior valor de F_3 optou-se por utilizar esta *template* em futuros ensaios e testes.

Analisando os resultados em maior detalhe (tabela 3.2) pode-se observar uma tendência para as transições suaves obterem piores resultados que as transições abruptas. Este facto sugere que é necessário empregar uma estratégia de *subsampling* de forma tornar o algoritmo mais sensível e, consequentemente, melhorar o valor de *Recall*.

Tabela 3.2: Resultados em detalhe para a *template* 25×25.

Tipo	T_p	F_n	Rec [%]
Wipe	17	16	51.5
Cut	714	162	81.5
Dissolve	353	333	51.4
Fade	2	6	33.3

3.4.2 Utilização de *subsampling*

De modo a minimizar o impacto das transições suaves na *performance* do algoritmo adotou-se uma estratégia de *subsampling* utilizando apenas uma em cada n *frames*.

Utilizando esta abordagem obtiveram-se os resultados apresentados na tabela 3.3.

Tabela 3.3: Resultados do algoritmo de segmentação utilizando *subsampling*.

n *frames*	T_p	F_n	F_p	Pre [%]	Rec [%]	F_3 [%]
1	1088	517	47	95.9	67.8	69.8
2	1331	274	95	93.3	82.9	83.8
3	1437	168	150	90.5	89.5	89.6

Como se pode constatar pela análise da tabela 3.4, utilizando apenas uma *frame* em cada três num vídeo permitiu aumentar o *recall* em cerca de 20% sacrificando apenas 5% de *precision*.

Em detalhe podemos observar também que o *recall* de transições suaves está

agora bastante mais próximo dos valores de *recall* das transições abruptas.

Tabela 3.4: Resultados em detalhe utilizando *subsampling* a 3 *frames*.

Tipo	T_p	F_n	Rec [%]
Wipe	28	5	84.8
Cut	819	57	93.5
Dissolve	103	333	85.0
Fade	6	3	66.7

Deste modo será utilizado um *subsampling* de 3 *frames* para os restantes testes e ensaios.

Explorou-se ainda a possibilidade de ignorar certas regiões de cada frame por forma a melhorar os resultados em termos de *recall*. Estas zonas geralmente não contribuem para o processo de segmentação por *shots*, isto é, que não apresentam alterações significativas durante as transições. Partindo deste princípio, a subsecção seguinte irá testar o impacto destas regiões para o processo de segmentação por *shots*.

3.4.3 Utilização de regiões

Na maioria dos blocos noticiosos existe uma barra na zona inferior da imagem, denominada oráculo, que contém informação extra sobre a notícia atual, assim como notícias de última hora. Apesar de ser importante do ponto de vista semântico pode ser considerada como irrelevante para a tarefa de segmentação por *shots* pois geralmente mantém-se igual de *shot* para *shot* (figura 3.5).

(a) Antes da transição (b) Depois da transição

Figura 3.5: Exemplo antes e depois de uma transição.

Observando a figura 3.5 é possível constatar que o oráculo mantém-se praticamente igual e que grande parte da complexidade (que seria detetada pela característica de contornos) está situada a meio da imagem e não na parte superior.

De modo a avaliar a importância da região de superior e inferior a imagem é dividida em três regiões, segundo o esquema apresentado na figura 3.6.

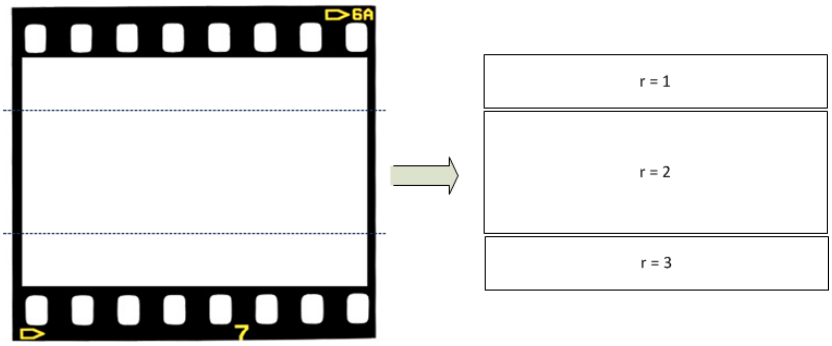


Figura 3.6: Regiões consideradas.

Nos ensaios efetuados definiram-se três cenários possíveis: o primeiro cenário (S1), as regiões r_1,r_2 e r_3 são consideradas como uma só região (r_{123}); o segundo cenário (S2), foram consideradas duas regiões a região composta pelas regiões r_1 e r_2 (r_{12}) e a região r_3; por fim, o cenário S3 contempla as três regiões individualmente.

A tabela 3.5 mostra os valores atribuídos ao p_r para cada cenário, note que p_r é a fração da imagem definida pela região r.

Tabela 3.5: Fração de cada região em cada um dos cenários.

Cenário	p_r
S1	$p_{123} = 1$
S3	$p_{12} = 0.875$ e $p_3 = 0.125$
S3	$p_{1,3} = 0.125$ e $p_2 = 0.75$

Para o cenário S1 obteve-se os seguintes resultados (Tabela 3.6).

No ensaio seguinte avaliámos a utilização de apenas duas regiões, conforme o descrito para o cenário S2, utilizando várias ponderações w_r. Os resultados destes

Tabela 3.6: Resultados do algoritmo de segmentação por *shots* para o cenário S1.

T_p	F_n	F_p	Rec [%]	Pre [%]	F_3 [%]
1436	149	164	89.5	90.5	89.6

ensaios no *corpus* de treino podem ser encontrados na tabela 3.7.

Tabela 3.7: Resultados do algoritmo de segmentação por *shots* para o cenário S2.

w_{12}	w_3	T_p	F_n	F_p	Rec [%]	Pre [%]	F_3 [%]
1,0	1,00	1436	149	164	89.5	90.5	89.6
1,0	0,00	1423	177	168	89.4	88.9	89.0

Observando as tabelas 3.6 e 3.7 não é possível notar uma melhoria dos resultados após a remoção por completo da região r_3. Para o ensaio com três regiões os resultados podem ser encontrados na tabela 3.8.

Tabela 3.8: Resultados do algoritmo de segmentação por *shots* para o cenário S3.

w_2	$w_{1,3}$	T_p	F_n	F_p	Rec [%]	Pre [%]	F_3 [%]
1,0	1,00	1436	149	164	89.5	90.5	89.6
1,0	0,00	1455	145	227	90.9	86.5	90.5

Comparando agora as tabelas 3.7 e 3.8 é possível concluir que ignorando a região superior se consegue obter algumas melhorias nos resultados em termos de *recall*. Isto pode ser explicado pelo facto de a região superior (r_1) não mudar significativamente em algumas das transições. Uma análise mais detalhada dos resultados da segmentação por *shots* pode ser encontrada na tabela 3.9.

Segundo a tabela 3.9 é ainda possível constatar que as transições suaves possuem um valor de *recall* menor (87.1%) que o conseguido nas transições abruptas (94.0%).

Em termos da relevância dos falsos negativos é possível constatar que apenas nove transições correspondiam ao início de histórias pelo que estes resultados dão

Tabela 3.9: Resultados do algoritmo de segmentação por *shots* por tipo de transição.

Tipo de Transição	Número	T_p	F_n	Rec [%]
Wipe	30	26	4	86.6
Cut	875	823	52	94.0
Dissolve	685	598	87	87.3
Fade	9	7	2	77.8

alguma confiança para o algoritmo de segmentação por histórias descrito no capítulo seguinte.

3.5 Conclusão

Neste capítulo apresentou-se um algoritmo de segmentação temporal de *shots*.

Como se pode constatar o algoritmo foi capaz de detetar grande parte das transições dos vídeos de treino. No entanto existe uma clara diferença entre os resultados para transições suaves e os resultados para transições abruptas.

As transições suaves são mais difíceis de detetar que as transições abruptas. Como tal foi necessário adotar estratégias adicionais para melhorar os resultados em termos de *recall*.

Uma dessas estratégias passa por ignorar dois em cada três *frames* do vídeo. Este facto fez com que se aumentasse a cobertura do algoritmo diminuindo ligeiramente a sua precisão.

A outra estratégia passou por excluir as regiões superiores e inferiores da *frame*. Ao fazer isto está-se a dar mais enfase na região central do vídeo, onde geralmente se passa a maior parte da ação.

Os resultados dos ensaios realizados demonstraram ainda que este algoritmo é uma base sólida para o algoritmo de segmentação por histórias dado que deteta uma elevada percentagem (cerca de 90%) das transições que iniciam novas histórias.

Capítulo 4

Algoritmo de segmentação por histórias

Neste capítulo descreve-se o algoritmo de segmentação por histórias. Desenvolvido especificamente para o caso dos programas noticiosos, esta metodologia assenta nas premissas desenvolvidas por Zhai e Goyal *et al* de que uma história começa geralmente por uma breve apresentação da peça por parte do pivô [23] [24] [25].

A análise de programas noticiosos portugueses permitiu constatar um conjunto de heuristicas associadas ao pivô. São estas: o pivô aparece geralmente sozinho; o pivô é a primeira pessoa a aparecer num programa noticioso, exceto quando existe algum tipo de introdução ou resumo do conteúdo; o pivô aparece regularmente ao longo de todo o segmento noticioso; o pivô apresenta uma consistência em termos de crominância e luminância; o pivô mantém geralmente a mesma distância à câmara ao longo de todo o programa. Dadas estas permissas torna-se necessário, numa primeira etapa da segmentação por histórias, identificar em que *keyframes* é possível encontrar o pivô.

A metodologia escolhida para o efeito é a comparação da crominância (utilizando histogramas normalizados) e a dimensão da face do pivô dado que estas características são relativamente constantes.

No entanto, o fundo da *keyframe* varia ao longo do segmento noticioso, como se pode verificar pela figura 4.1, pelo que este deve ser removido no processo antes do cálculo dos histogramas. Por outro lado, a cor do vestuário do pivô mantém-se ao longo de todo o programa pelo que se considerou que esta é a zona ideal para a extração das características.

Figura 4.1: Exemplos do pivô ao longo de um programa noticioso (TJ_{tr2}).

Como o algoritmo funciona em tempo real, a estratégia usada fica condicionada pelo facto de não se ter acesso, em diferido, à totalidade do programa o que exclui a análise de ciclos e muitos algoritmos de *clustering*. A solução encontrada passa por comparar cada *keyframe* a uma *template* (denominada de *template-pivô*) que é criada de forma automática para cada programa noticioso.

Partindo do princípio que o pivô é a primeira pessoa a aparecer num qualquer segmento noticioso pode-se concluir que a primeira *keyframe* com face é uma boa candidata a *template-pivô*. Deste modo, numa primeira etapa, as *keyframes* são analisadas por forma a definir a *template-pivô* (secção 4.2) através da aplicação do algoritmo Viola-Jones para a deteção facial.

A deteção do pivô consistirá no cálculo de duas características, nomeadamente a diferença de histogramas e área da face relativamente à área da face da *template-pivô* (secção 4.3), e na sua deteção através de um classificador treinado (secção 4.4).

Por fim, todos estes passos serão optimizados na secção 5.4 por forma a garantir os melhores resultados possiveis para os testes efetuados no capítulo seguinte.

Todos estes passos podem ser resumidos segundo o fluxograma presente na figura 4.2.

Uma vez que a deteção facial é um passo transversal ao algoritmo de segmen-

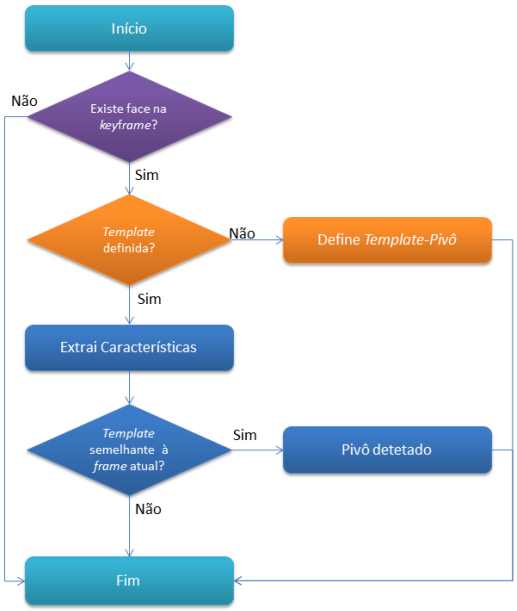


Figura 4.2: Fluxograma do algoritmo de segmentação por histórias.

tação por histórias (isto é, é utilizada tanto na seleção da *template-pivô* como na extração de características em cada *keyframe*), na secção 4.1, ir-se-á proceder a uma explicação detalhada da metodologia utilizada para o efeito.

4.1 Deteção facial

Durante todo o processo de segmentação por histórias as *keyframes* são analisadas de modo a verificar a presença do pivô na imagem. Partindo do princípio que um pivô aparece geralmente em *shots* que possuam uma única face, é necessário proceder à deteção da faces presentes em cada *keyframe*.

Para este efeito, utilizou-se uma ferramenta disponibilizada no Matlab 2012a. Esta ferramenta faz parte da Image Processing Toolbox e utiliza o algoritmo de deteção de objetos Viola-Jones [4].

Para além da deteção facial este algoritmo permite a deteção de outros objetos como, por exemplo, olhos, face em perfil, boca, nariz e tronco superior (tórax, pescoço e cabeça).

Uma das razões pelo qual este algoritmo foi escolhido é o facto de ser, segundo os autores, bastante rápido, sendo possível a sua execução a 15 fps num computador Pentium III a 700 MHz [4]. Deste modo não irá prejudicar a *performance*, em termos de velocidade de processamento, do algoritmo de segmentação por histórias.

O algoritmo Viola-Jones desenvolve-se em 3 etapas. Na primeira etapa a imagem é transformada numa representação intermédia denominada *integral image*. Nesta imagem, o valor de um píxel numa *integral image* é calculado através da soma do valor dos píxeis acima e à esquerda, segundo a equação 4.1 [4].

$$ii(x,y) = \sum_{x'=0,y'=0}^{x'=x,y'=y} i(x',y') \tag{4.1}$$

onde ii é a *integral image* e i a imagem original.

Sobre esta representação intermédia da imagem serão calculadas várias características tipo Haar. Estas características, envolvem a soma do valor dos píxeis da imagem em áreas retangulares, pelo que se assemelham a uma transformada de Haar (proposta por Alfred Haar em 1909). Algumas das possíveis características podem ser observadas na figura 4.3 [4].

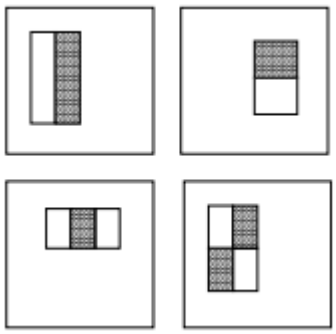


Figura 4.3: Exemplo de características Haar (retirado de [4]).

O cálculo destas características é bastante mais rápido numa *integral image* dado que esta já resulta de uma soma cumulativa.

A última etapa utiliza estas características e, recorrendo a um conjunto de classificadores em cascata (nomeadamente perceptrões), procede à deteção do objeto.

O facto de os classificadores mais complexos serem apenas aplicados depois dos classificadores mais simples (e rápidos) torna o algoritmo computacionalmente mais eficiente, visto que apenas um subconjunto das características será avaliada por todos os classificadores.

Em qualquer imagem o número de características possíveis é bastante elevado (cerca de 180 mil), como tal, o algoritmo utiliza o método AdaBoost para assegurar uma classificação rápida na etapa seguinte ao eliminar características menos relevantes para o objeto a detetar [4].

Para o algoritmo de segmentação por histórias a deteção facial é feita em duas etapas. Em primeiro lugar o algoritmo Viola-Jones é utilizado de forma a procurar na imagem o tronco superior (tórax, pescoço e cabeça) do pivô. Posteriormente, por cada tronco encontrado será feita uma pesquisa pela face frontal. Deste modo reduz-se a possibilidade de falsos positivos dado que a face apenas é considerada após a deteção do tronco.

A diminuição de falsos positivos tem como contrapartida a diminuição da velocidade de processamento em cada *keyframe* dado que duas buscas são realizadas em vez de apenas uma.

No processo de deteção de pivôs a deteção facial terá um papel vital pois é utilizada não só para a remoção do fundo de cada *keyframe* mas também para a seleção da *template-pivô*. A seleção desta *template* será tratada na secção seguinte.

4.2 Seleção da *template-pivô*

A primeira etapa do algoritmo de segmentação por histórias é a seleção de uma *template-pivô*.

Esta *template* será a base com a qual todas as outras *keyframes* candidatas serão comparadas. Como tal, a condição para a seleção de uma *template* válida é a de que esta deve ter presente o pivô.

A *template-pivô* poderia ser seleccionada manualmente, no entanto considerou-se que essa solução não é prática pelo que a seleção será feita automaticamente.

Sabendo que o pivô é geralmente a primeira pessoa a aparecer num segmento noticioso, pode-se considerar que a primeira *keyframe* a conter uma pessoa é geralmente um forte candidato à *template-pivô*. Desta forma considerou-se que a *template-pivô* é a primeira *keyframe* a apresentar uma face.

Após a definição da *template-pivô* serão extraídas uma série de características para a deteção do pivô. Estas características serão descritas em maior detalhe na secção seguinte.

4.3 Características para a deteção de pivô

Na segunda fase do algoritmo procede-se à comparação de cada *keyframe* com a *template-pivô* selecionada na etapa anterior.

Através da observação de vários programas noticiosos, e indo ao encontro da investigação de Zhai *et al*, podemos observar que um pivô mantém a mesma roupa ao longo de todo o segmento noticioso [23]. Esta é portanto uma pista visual bastante importante para a segmentação de um qualquer programa noticioso.

Para o cálculo das características nem toda a área da imagem é útil. Como tal, é necessário, em primeiro lugar, excluir o fundo da imagem através de uma máscara.

4.3.1 Definição da máscara

Durante todo o segmento noticioso a característica que se vai mantendo é a cor do fato do pivô, pelo que a inclusão do fundo da imagem no cálculo dos histogramas, só iria piorar os resultados da segmentação.

Como tal, Zhai *et al* propõe que a área utilizada na comparação deva ser calculada através do prolongamento da zona da face para baixo, de modo a incluir alguma parte do vestuário do pivô (observe a figura 4.4) [23]. Por si só este procedimento evita quase todo o ruído, dado que exclui quase por completo, o fundo da imagem. Esta técnica desperdiça, no entanto, grande parte da área do vestuário.

Para a deteção de pivô propõe-se a utilização de dados estatísticos antropo-

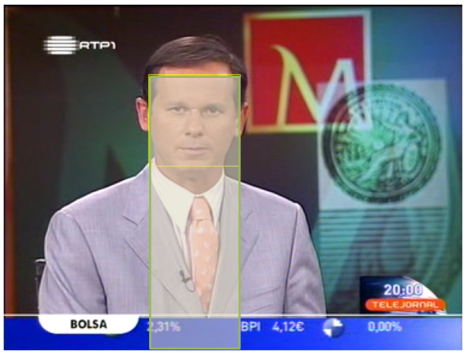


Figura 4.4: Zona da imagem considerada por Zhai.

métricos de modo a estimar uma zona abaixo da face que seja considerada fato. A antropometria é definida como o conjunto de técnicas utilizadas para medir o corpo humano ou as suas partes. Deste modo, dados antropométricos estatísticos, podem fornecer uma estimativa razoável das dimensões típicas do corpo humano e respetivas proporções.

Após algumas experiências, optou-se por utilizar dados antropométricos, em detrimento do tronco superior detetado pelo algoritmo Viola-Jones, por se considerar que os resultados deste incluiam ainda parte do fundo da frame.

A National Aeronautics and Space Administration (NASA) disponibiliza alguns dados antropométricos típicos relativos às medidas especificadas na figura 4.5. No entando, nem todas estas dimensões são relevantes para esta etapa do trabalho, e como tal consideram-se apenas as medidas da largura da face (*Head breadth*) e a largura de ombros (*Bideltoid breadth*).

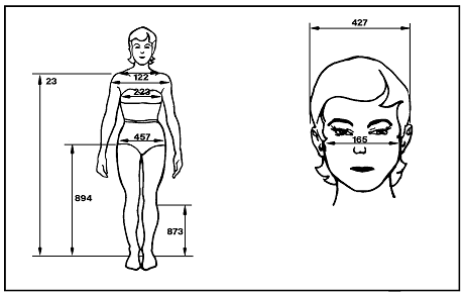


Figura 4.5: Alguns dados antropométricos (retirados de [5]).

Os valores típicos para estas medidas podem ser encontrados na tabela 4.1 [5].

Tabela 4.1: Dados antropométricos (retirados de [5]).

Legenda na Imagem	Denominação	Homem	Mulher
165	*Head breadth*	15,7 cm	15,6 cm
122	*Bideltoid breadth*	48,9 cm	38,9 cm

Como se pode constatar as medidas 165 e 122 têm correspondência com as variáveis a e b da figura 4.6.

Figura 4.6: Zonas consideradas na comparação com a *template*.

Deste modo, sabendo a largura em píxeis da face torna-se possível estimar a largura de ombros dado que estes são 2,5 a 3,1 vezes mais largos que a largura da face. Por forma a evitar a inclusão de parte do fundo da imagem escolheu-se o menos destes valores para a estimação da largura de ombros.

Sendo estes valores relativos não são afetados pela distância a que o sujeito se encontra da câmara, isto é, utilizando a equação 4.2, é possível calcular a distância b a partir de a (que será calculado através do algoritmo Viola-Jones) .

$$b = 2.5 \times a \tag{4.2}$$

Para além da máscara que irá indicar quais os píxeis a ser considerados no cálculo dos histogramas este passo fornece ainda as dimensões da face do pivô.

Estas informações são importantes dado que o pivô se mantem a uma distância relativamente constante à câmara e, como tal, a área da face será também um

indicador a considerar durante a classificação.

Sendo a crominância do fato do pivô constante ao longo do programa noticioso, e como tal um indicador importante, é necessário comparar os histogramas da *template-pivô* e da *keyframe*. O método escolhido para a comparação é a diferença euclidiana de histogramas cumulativos, descrito na subsecção seguinte.

4.3.2 Cálculo do histograma cumulativo

Após a obtenção dos píxeis a utilizar no processo de comparação, são calculados os histogramas normalizados da *template* e do *keyframe* atual. Estes histogramas são calculados no espaço de cor HSV, em detrimento do espaço de cor RGB, pois HSV é geralmente mais utilizado quando se pretende comparar cor.

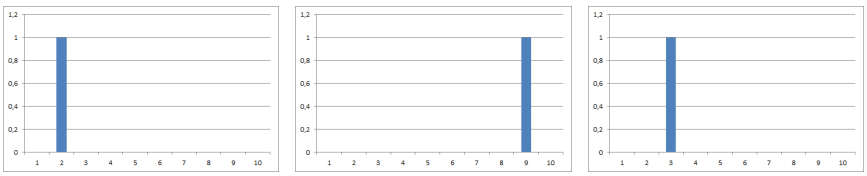
Isto acontece porque o espaço de cor HSV possui uma componente dedicada à matiz, outra à saturação e outra à luminusidade enquanto que o espaço de cor RGB "mistura" estas três propriedades da cor nas suas componentes.

Em alternativa à utilização da diferença euclidiana "simples" dos histogramas normalizados este trabalho propõe que se faça a diferença entre histogramas cumulativos. Isto porque se considera que a diferença entre histogramas definida na equação 4.3 não reflete no seu valor pequenas variações na intensidade média da imagem.

$$d\left(f_{i-1}, f_i\right) = \sqrt{\sum_{j=0}^{j<B} \sum_{k=0}^{k<3} \left[H_{i-1}\left(j,k\right) - H_i\left(j,k\right)\right]^2} \tag{4.3}$$

Tomando como exemplo os histogramas da figura 4.7, podemos concluir que a distância euclidiana entre os histogramas A e B é igual à diferença euclidiana entre os histogramas A e C, apesar destes dois últimos serem mais semelhantes em termos de intensidade.

Ao realizar a soma cumulativa dos histogramas, e após o cálculo da distância euclidiana, é possível observar que a distância entre os histogramas A e B é igual a 6 e a distância entre os histogramas A e C é igual a 1. O que é uma representação mais fiel do que se passa ao nível da imagem.



(a) Histograma A (b) Histograma B (c) Histograma C

Figura 4.7: Exemplo da utilização de histogramas cumulativos na comparação de imagens.

Deste modo, para o cálculo desta característica, definiu-se a função Histograma como sendo a seguinte (equação 4.4):

$$H_i = h_i + H_{i-1} \tag{4.4}$$

Onde h_i é o número de píxeis com o valor i a dividir pelo número total de píxeis a considerar.

O valor de d (equações 4.3 e 4.4), em conjunto com a área da face (em relação à área da face presente na *template*) será passado a um classificador Naive-Bayes. Este aspeto será tratado em maior detalhe na secção seguinte.

4.4 Deteção de pivô

De modo a detetar a presença de um pivô na *keyframe* atual utilizaram-se a área da face e a distância entre histogramas normalizados (descritos na secção anterior).

Estes dados serão depois enviados a um classificador do tipo Naive-Bayes que, tendo em conta os dados recolhidos no conjunto de treino, irá classificar os dados extraídos numa de duas categorias possíveis: "Pivô Presente" ou "Pivô Ausente", P_p ou P_a, respetivamente.

Quando uma *keyframe* é classificada como pertencente à categoria "Pivô Presente" então será assinalada o início de nova história.

Este classificador, baseado no teorema de Bayes (equação 4.5), começa por assu-

mir que todas as características recolhidos são independentes, isto é, o classificador assume que a área da face do pivô não depende da distância entre histogramas e vice-versa.

$$P\left(C|F_1,...,F_n\right) = \frac{P\left(F_1,...,F_n|C\right)\cdot P\left(C\right)}{P\left(F_1,...,F_n\right)} \tag{4.5}$$

Onde $F_1,...,F_n$ são as características possíveis e C a classe.

Como se considera que todas as características são independentes, então é possível fazer a simplificação presente na equação 4.6.

$$P\left(F_1,...,F_n|C\right) = \frac{P\left(C\right)\prod_{k=1}^{n} P\left(F_k|C\right)}{P\left(F_1,...,F_n\right)} \tag{4.6}$$

Para exemplificar o processo de classificação considere-se os dados da tabela 4.2.

Tabela 4.2: Cálculo dos paramêtros para o classificador Naive-Bayes, retirados do *corpus* de treino.

Classe	Área da face (af)		Distância entre histogramas (dh)	
	Média (μ_{af})	Desvio Padrão (σ_{af})	Média (μ_{dh})	Desvio Padrão (σ_{dh})
P_p	-0.035	0.231	49.374	22.547
P_a	-0.046	0.484	142.119	56.542

Para além dos dados presentes na tabela 4.2, e utilizando os dados do *corpus* de treino, foi ainda possivel calcular que $P(P_p) = 0.317$ e $P(P_a) = 0.683$.

Supondo então que é necessário classificar uma nova *keyframe* com uma área de face igual a 0.847 da área da face da *template-pivô* e uma distância entre histogramas igual a 126.994.

Calcula-se agora as probabilidades condicionadas para a nova *keyframe*, utilizando para o efeito a função densidade de uma distribuição normal definida com os parâmetros da tabela 4.2 (equações 4.7 , 4.8 , 4.9 e 4.10).

$$\begin{aligned} P\left(af = 0.84681|P_p\right) &= \frac{1}{\sqrt{2\cdot\pi\cdot(\sigma_{af})^2}} exp(\frac{-(af-\mu_{af})^2}{2\cdot(\sigma_{af})^2}) \\ &= \frac{1}{\sqrt{2\cdot\pi\cdot(0.2309)^2}} exp(\frac{-(0.84681+0.0349)^2}{2\cdot(0.2309)^2}) \\ &= 0.0012 \end{aligned} \tag{4.7}$$

$$\begin{aligned} P\left(d = 126.993|P_p\right) &= \frac{1}{\sqrt{2\cdot\pi\cdot(\sigma_{dh})^2}} exp(\frac{-(dh-\mu_{d})^2}{2\cdot(\sigma_{dh})^2}) \\ &= \frac{1}{\sqrt{2\cdot\pi\cdot(22.5471)^2}} exp(\frac{-(126.993-49.3738)^2}{2\cdot(22.5471)^2}) \\ &= 0.000047 \end{aligned} \tag{4.8}$$

$$\begin{aligned} P\left(af = 0.84681|P_a\right) &= \frac{1}{\sqrt{2\cdot\pi\cdot(\sigma_{af})^2}} exp(\frac{-(af-\mu_{af})^2}{2\cdot(\sigma_{af})^2}) \\ &= \frac{1}{\sqrt{2\cdot\pi\cdot(0.4837)^2}} exp(\frac{-(0.84681+0.0460)^2}{2\cdot(0.4837)^2}) \\ &= 0.2095 \end{aligned} \tag{4.9}$$

$$\begin{aligned} P\left(d = 126.993|P_a\right) &= \frac{1}{\sqrt{2\cdot\pi\cdot(\sigma_{dh})^2}} exp(\frac{-(dh-\mu_{dh})^2}{2\cdot(\sigma_{dh})^2}) \\ &= \frac{1}{\sqrt{2\cdot\pi\cdot(56.5424)^2}} exp(\frac{-(126.993-142.1188)^2}{2\cdot(56.5424)^2}) \\ &= 0.0068 \end{aligned} \tag{4.10}$$

Utilizando estes valores na equação 4.6 obtém-se os valores das equações 4.11 e 4.12, que correspondem às probabilidades da nova *keyframe* pertencer à classe "Pivô Presente" e "Pivô Ausente", respectivamente:

$$\begin{aligned} P\left(P_p|af, dh\right) &= \frac{P(P_p)\cdot P(af|P_p)\cdot P(dh|P_p)}{P(P_p)\cdot P(af|P_p)\cdot P(dh|P_p)+P(P_a)\cdot P(af|P_a)\cdot P(dh|P_a)} \\ &= 1.8409 \times 10^{-5} \end{aligned} \tag{4.11}$$

$$\begin{aligned} P\left(P_a|af, dh\right) &= \frac{P(P_a)\cdot P(af|P_a)\cdot P(dh|P_a)}{P(P_p)\cdot P(af|P_p)\cdot P(dh|P_p)+P(P_a)\cdot P(af|P_a)\cdot P(dh|P_a)} \\ &\approx 1 \end{aligned} \tag{4.12}$$

Como $P\left(P_a|af, dh\right) > P\left(P_p|af, dh\right)$ ir-se-ia classificar esta *keyframe* como pertencendo à classe "Pivô Ausente".

A utilização de classificadores treinados, em detrimento de outras técnicas como a análise de *loops* ou algoritmos de agrupamento, permite que a segmentação por histórias seja feita em tempo real, e à medida que as *frames* são processadas.

Deste modo conclui-se o processo de deteção de pivô e consequentemente o algoritmo de deteção de histórias implementado. No entanto antes de se proceder aos testes e ensaios finais é necessário testar cada um dos componentes do algoritmo por forma a otimizar o seu desempenho.

4.5 Otimização do algoritmo de segmentação por histórias

De forma a avaliar o algoritmo de segmentação por histórias realizaram-se várias experiências. Estas experiências tiveram como objetivo verificar a correção as escolhas tomadas no desenvolvimento do algoritmo. Utilizaram-se, para o efeito, os resultados da segmentação automática por *shots* excluindo os *shots* (onde estava presente o pivô) cujas transições não foram corretamente identificadas.

Deste modo, o procedimento para a avaliação do algoritmo de segmentação por histórias passa por avaliar cada componente do algoritmo (nomeadamente histogramas, espaço de cor e tamanho/forma da máscara) individualmente e verificar se as escolhas feitas neste capítulo trouxeram algum tipo de benefício em termos de *precision* e *recall*.

A primeira dessas escolhas passa pela utilização de um histograma cumulativo normalizado em detrimento de um histograma "normal". Os resultados que comparam ambas as abordagens podem ser consultados na subsecção seguinte.

4.5.1 Histograma cumulativo

Nesta subsecção apresentam-se os resultados da aplicação do algoritmo de segmentação por histórias sobre o conjunto de treino de modo a aferir se a utilização de um histograma cumulativo tem algum benefício em comparação com a utilização de um histograma "normal".

Os resultados dos ensaios podem-se encontrar na tabela 4.3. Estes valores foram obtidos utilizando todo o fato para a produção de histograma (no espaço de cor HSV) e o classificador treinado de Bayes.

Tabela 4.3: Resultados do algoritmo de segmentação por histórias utilizando histograma cumulativo.

Tipo de Histograma	T_p	F_n	F_p	Pre [%]	Rec [%]	F_3 [%]
Normal	60	23	18	76.9	72.2	72.7
Cumulativo	69	14	25	79.3	83.1	82.7

Tal como seria de esperar, a utilização de histogramas cumulativos produz melhores resultados, tanto em termos de *precision* como de *recall*.

Apesar de uma melhoria nos resultados, face ao histograma "normal", a utilização do histograma cumulativo não é o único aspeto que foi trabalhado no algoritmo de segmentação por histórias. Como tal, a definição da forma da máscara será tratada na subsecção seguinte.

4.5.2 Definição da máscara

Outro ponto que necessita de ser avaliado é a utilização de todo o corpo do pivô *versus* a utilização da parte central.

Os resultados da segmentação podem ser visualizados na tabela 4.4. Utilizando agora o histograma cumulativo é possível observar que a utilização de todo o corpo do pivô apresenta algumas melhorias em relação a uma implementação onde a maior parte do fato é ignorado.

Tabela 4.4: Resultados do algoritmo de segmentação por histórias utilizando o fato completo.

Tipo de Mascara	T_p	F_n	F_p	Pre [%]	Rec [%]	F_3 [%]
Apenas zona central	65	18	24	73.0	78.3	77.7
Fato completo	69	14	25	73.4	83.1	82.0

Por último resta verificar se com a utilização do espaço de cor HSV, em detrimento do espaço de cor RGB, é possível melhorar os resultados em termos de

precision e *recall*. Para tal, foram efetuados os ensaios descritos na subsecção seguinte.

4.5.3 Espaço de cor

Relativamente ao espaço de cor, utilizado no cálculo dos histogramas, decidiu-se que HSV é capaz de produzir melhores resultados, porque a tonalidade da cor é definida em apenas um dos seus três componentes. De modo a validar esta decisão decidiu-se testar o algoritmo de segmentação por histórias utilizando RGB e HSV. Os resultados destes ensaios estão presentes na tabela 4.5.

Tabela 4.5: Resultados do algoritmo de segmentação de historias utilizando o espaço de cor HSV.

Espaço de Cor	T_p	F_n	F_p	Pre [%]	Rec [%]	F_3[%]
RGB	69	14	25	73.4	83.1	82.0
HSV	69	14	18	79.3	83.1	82.7

Estes resultados demonstram que HSV é o espaço de cor mais indicado (quando comparado com RGB) para a realização do processo de segmentação por histórias.

Deste modo, e tendo-se já definido e testado cada uma das outras componentes do algoritmo, é possível proceder aos testes finais sobre o *corpus* de teste no capítulo seguinte.

4.6 Conclusão

Neste capítulo introduziu-se um novo algoritmo de segmentação por histórias baseado na deteção do pivô.

A deteção é efetuada utilizando várias métricas que resultam da comparação da *keyframe* candidata à *template* escolhida. Estas características são depois classificadas por um classificador Naive-Bayes.

Todo este processo permite que a segmentação por histórias seja feito em tempo

real à medida que as *keyframes* vão sendo escolhidas.

Por forma a testar a validade do algoritmo quando comparado com outras abordagens foram efetuadas várias experiências.

Na primeira dessas experiências testou-se a utilização de histogramas cumulativos em detrimento de um histograma "normal". Tal como seria de esperar o histograma cumulativo obteve melhores resultados.

Na segunda experiência testou-se a utilização de todo o fato do pivô *vs* a utilização de apenas a parte central do fato (como é referido no levantamento do estado da arte). A utilização de todo o fato do pivô mostrou melhores resultados que a utilização de apenas a parte central.

Por último comparou-se os resultados para a utilização do espaço de cor HSV em detrimento do mais comum RGB. O Espaço de cor HSV demonstrou ser o mais adequado para este tipo de aplicações possuindo melhores resultados em termos de precisão.

Capítulo 5

Testes e resultados

Neste capítulo decrevem-se as experiências realizadas com o objetivo de avaliar a *performance* dos algoritmos de segmentação por *shots* e por histórias, sendo utilizado, para o efeito, o *corpus* de teste.

Os resultados destes testes serão quantificados através das métricas de avaliação definidas na secção 5.1. Estas métricas serão calculadas segundo a metodologia presente na secção 5.2.

Os ensaios realizados no *corpus* de teste serão descritos, para o caso do algoritmo de segmentação por *shots*, na secção 3.4. Para a segmentação ao nível da história os resultados serão apresentados na secção 5.4.

Por último apresentam-se os resultados, em termos de velocidade de processamento, de ambos os algoritmos na secção 5.5.

5.1 Métricas de avaliação

Por forma a quantificar e comparar a *performance* dos algoritmos desenvolvidos utilizar-se-ão métricas como a *precision*, *recall* e *F-Measure*.

Estas métricas são baseadas, por sua vez, a partir de um conjunto de métricas (denominadas de métricas base) que quantificam diretamente os eventos (*hits*, *miss* e falsos alarmes) que ocorrem ao longo do processo de segmentação.

Na subsecção seguinte irá-se explorar as métricas base em maior detalhe.

5.1.1 Métricas base

As métricas base quantificam diretamente o número de ocorrências de eventos significativos ao processo de segmentação, podendo ser classificados em deteções corretas, incorretas e não-deteções. Estes serão a base para o cálculo de métricas como *precision* e *recall*. As três métricas utilizadas são *True Positives* (Tp), *False Positives* (Fp) e *False Negatives* (Fn).

Verdadeiros positivos (Tp, do inglês *true positives*, ou *hits*), são os segmentos que foram corretamente identificados pelo algoritmo de segmentação.

Os falsos positivos (Fp), são os falsos alarmes. Indicam que o algoritmo detetou, incorretamente, a presença de um novo segmento ou transição.

Falsos negativos (Fn) são os segmentos que deveriam de ter sido identificados pelo algoritmo e não o foram, podendo também ser descritos como *misses*.

Por muito úteis que estas métricas sejam não permitem comparar a *performance* dos algoritmos de segmentação em vídeos com dimensão (número de *shots*) diferentes. Para tal, é necessário aplicar métricas que utilizem estes valores sob a forma de percentagem. Uma das métricas que é geralmente utilizada para o efeito é a *Precision*.

5.1.2 *Precision*

Precision (Pre), é neste contexto a percentagem de transições que tendo sido identificadas pelo algoritmo, correspondem na realidade a transições no programa, como se pode ver na equação 5.1.

$$Pre = \frac{Tp}{(Tp + Fp)} \times 100 \tag{5.1}$$

Apesar desta métrica quantificar corretamente a precisão dos algoritmos desenvolvidos, não entra em conta com o número de eventos que deveriam ter sido detetados e não o foram (os falsos negativos). Na subsecção seguinte será descrita uma métrica que complementa a *precision* neste aspeto.

5.1.3 *Recall*

Recall (Rec) é definido como sendo a percentagem de transições que existindo no programa, são corretamente identificadas. Este valor é muitas vezes utilizado como uma medida de cobertura do algoritmo e é definida segundo a equação 5.2.

$$Rec = \frac{Tp}{(Tp + Fn)} \times 100 \tag{5.2}$$

Com a *precision* e o *recall* é possível ter uma ideia da *performance* dos algoritmos de segmentação. No entanto, ainda não é possível comparar dois algoritmos simultaneamente em termos de precisão e cobertura. Para tal é utilizada uma métrica que engloba tanto os valores de *Precision* como os valores de *Recall* denominada *F-measure*.

5.1.4 *F-measure*

A média harmónica ponderada (*F-measure*) é geralmente utilizada para comparar algoritmos, dado que agrupa as métricas de *precision* e *recall* numa única [7]. A importância desta metrica é o facto de permitir dar mais peso ao valor da *Precision*

ou *Recall* consoante se pretenda que um falso positivo tenha mais impacto que um falso negativo, ou vice-versa.

Esta métrica é definida de acordo com a equação 5.3.

$$F_\beta = \frac{(1+\beta^2)\cdot Pre \cdot Rec}{\beta^2 \cdot Pre + Rec} \qquad \forall\ \beta \in \mathbb{R}_+^0 \tag{5.3}$$

Na equação 5.3 o *Recall* é β vezes mais importante que a precisão, deste modo é possível igualar β a 1 quando se pretende que ambos os parâmetros tenham igual peso, obtendo assim a equação 5.4.

$$F_1 = 2 \cdot \frac{Pre \cdot Rec}{Pre + Rec} \tag{5.4}$$

No entanto, no processo de identificação de pivôs (e segmentação por histórias) é mais crítico não detetar uma transição do que ter um falso alarme. Por esse motivo considerou-se que o valor de *Recall* é três vezes mais importante que a precisão do algoritmo dando origem assim à métrica F_3, dado que $\beta = 3$.

Para a obtenção destas métricas é necessário portanto processar os resultados dos algoritmos de segmentação e compará-los com os resultados da segmentação manual. Esse processo será descrito em maior detalhe na secção seguinte.

5.2 Metodologia de avaliação

De forma a ser possível gerar as métricas propostas na secção anterior é necessário comparar os resultados da segmentação automática (efetuada através dos algoritmos desenvolvidos nos capítulos anteriores) e os resultados da segmentação manual. O processo poderá ser esquematizado segundo a figura 5.1.

Para comparar os resultados da segmentação manual com a segmentação automática, desenvolveu-se um *script* de comandos Linux (do tipo *bash shell script*) que tem como principal função contabilizar o falsos positivos, falsos negativos e verdadeiros positivos (Anexo A).

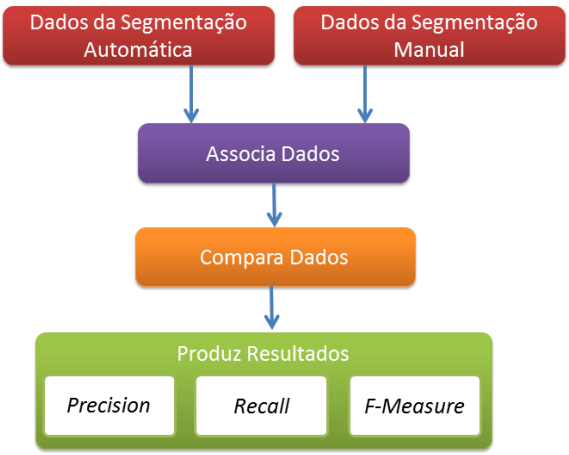


Figura 5.1: Diagrama do processo de avaliação.

Com base neste conjunto de metodologias e métricas é possível proceder ao treino e avaliação dos algoritmos de segmentação. Na secção seguinte será feita a otimização e teste do algoritmo de segmentação por *shots*.

5.3 Resultados para a segmentação por *shots*

Utilizando os parâmetros determinados anteriormente, isto é, com uma *template* quadrada 25×25, *subsampling* de 3 *frames* e ignorando as regiões superiores e inferiores da *frame*, aplicou-se o algoritmo de segmentação em *shots*. Os resultados podem ser visualizados nas tabelas seguintes (5.1 e 5.2).

Tabela 5.1: Resultados para a segmentação por *shots*.

T_p	F_n	F_p	Pre [%]	Rec [%]	F_3 [%]
2082	199	251	89.2	91.3	91.1

Comparando os resultados deste trabalho com os resultados obtidos em 2005

Tabela 5.2: Resultados em detalhe para a segmentação por *shots*.

Tipo	T_p	F_n	Rec [%]
Wipe	52	2	96.3
Cut	695	39	94.7
Dissolve	1307	157	89.3
Fade	24	1	96.0

no TRECVID (tabela 5.3) podemos verificar que este trabalho apresentou melhores resultados. Esta comparação serve apenas para dar algum contexto visto que os conjuntos de teste e treino utilizados não são os mesmos.

Tabela 5.3: Resultados para o *Recall* comparativamente ao TRECVID 2005 [32].

Tipo	TRECVID	Trabalho
Transições Abruptas	93.6 %	94.7 %
Transições Suaves	78.8 %	89.6 %

Com estes resultados, e de acordo com a pesquisa bibliográfica efetuada, pode-se concluir que, apesar dos resultados serem satisfatórios na sua generalidade, as transições suaves são geralmente mais difíceis de detetar, pelo que possuem um menor valor de *recall*.

Como este algoritmo será a base para a segmentação por histórias é crítico que este tenha um bom desempenho em termos de *recall*. Note que se considerou mais importante a sua cobertura do algoritmo que a precisão (daí a escolha da métrica F_3 para a avaliação do algoritmo), dado que iria influenciar os resultados do algoritmo de segmentação por histórias.

Contabilizando o número de falsos negativos ocorridos em shots com pivô é possível verificar que apenas 10 dos falsos negativos detetados corresponderem ao início de uma nova história. Considera-se então que este algoritmo pode servir de base ao algoritmo de segmentação por histórias.

5.4 Resultados para a segmentação por histórias

Após a definição de todos os parâmetros do algoritmo de segmentação por histórias (espaço de cor, forma da máscara e tipo de histograma) foram realizados os ensaios sobre o *corpus* de teste. Os resultados podem ser consultados na tabela 5.4.

Estes resultados foram obtidos após a segmentação automática pelo algoritmo de segmentação por *shots* e excluindo os falsos negativos e falsos positivos por este provocados.

Tabela 5.4: Resultados finais do algoritmo de segmentação por histórias.

T_p	F_n	F_p	Pre [%]	Rec [%]	F_3 [%]
85	34	14	85.8	71.4	72.6

Comparando com o estado da arte este trabalho obteve um valor para a métrica F_1 igual a 77.9 %, o que comparado com o TRECVID 2004, onde se obtiveram resultados de 69 %, revela que esta técnica de deteção de pivô é bastante promissora em termos de poder vir a proporcionar uma ferramenta automática de segmentação temporal de programas noticiosos [33].

Apesar destes resultados não foi possível ainda verificar a eficiência, em termos de tempo de processamento, dos algoritmos desenvolvidos. Esta vertente da avaliação será explorada na secção seguinte.

5.5 Performance do algoritmo

Em termos de velocidade de processamento o algoritmo foi capaz de processar 54 minutos de vídeo em 29 minutos (53.7%) num computador com as características presentes na tabela 5.5.

Note-se ainda que grande parte do tempo de processamento (14.6% do tempo do vídeo) foi gasto na leitura das *frames* a partir do disco rígido. Desprezando o tempo de acesso aos dados, e supondo que as *frames* iriam chegar a partir de uma

Tabela 5.5: Computador utilizado nos ensaios de performance.

Característica	Valor
Sistema Operativo	Windows 7 - Home Premium
Processador	Intel T6600 (2.20 GHz e 64 bits)
Memória RAM	4 GB
Disco Rígido	Fujitsu MJA2500BH-G2 (SATA 3.0Gb/s e 5400 RPM)

fonte externa (placa de captura de vídeo), pode-se afirmar que o algoritmo seria capaz de processar o vídeo completo em apenas 21 minutos (cerca de 39.1% da duração do programa).

Os resultados detalhados para a segmentação de *shots* encontram-se na tabela 5.6 e na figura 5.2, sendo que esta fase do algoritmo demora apenas 14 minutos, ou seja 27.4% da duração do programa.

Tabela 5.6: Análise da *performance* do algoritmo de segmentação por *shots*.

Etapa	Tempo [min]	Percentagem [%]
Calcular contornos	7.0	47.7
Espessamento de contornos	4.5	30.5
Conversão para escala de cinza	1.4	9.2
Outras	1.9	12.6

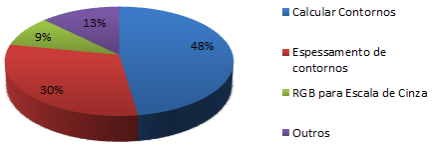


Figura 5.2: Análise da *performance* do algoritmo de segmentação por *shots*.

Os restantes 7 minutos do tempo de processamento (ou seja, 11.7 % da duração do programa) foram utilizados pelo algoritmo de segmentação por histórias. Pode-se visualizar na tabela 5.7 e na figura 5.3 que o algoritmo responsável pela localização

de faces na imagem é o grande responsável pelo tempo de processamento requerido pelo algoritmo de segmentação por histórias.

Tabela 5.7: Análise da *performance* do algoritmo de segmentação por histórias.

Etapa	Tempo [s]	Percentagem [%]
Localizar faces	349,13	92
Histograma seletivo	28,69	7,6
Classificação	1,60	0,4

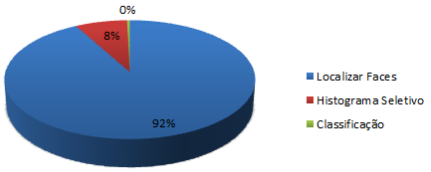


Figura 5.3: Análise da *performance* do algoritmo de segmentação por histórias.

5.6 Conclusão

Neste capítulo descreveram-se as várias experiências realizadas com o objetivo de aferir o desemprenho dos algoritmos desenvolvidos, nomeadamente, o algoritmo de segmentação por *shots* e o algoritmo de segmentação por histórias.

Por forma a quantificar os resultados definiram-se as métricas de avaliação *Precision*, *Recall* e *F-measure*. A *performance* computacional foi também analisáda por forma a aferir se os algoritmos desenvolvidos eram capazes de processar um vídeo em tempo real.

A discussão e reflexão sobre estes dados será realizada no próximo capítulo.

Capítulo 6

Conclusões e trabalho futuro

Neste documento descreveu-se um sistema automático de segmentação por *shots* e por histórias de programas noticiosos em suporte de vídeo.

O sistema desenvolvido apresenta como resultado do processamento, um ficheiro de anotação compatível com a ferramenta de anotação Anvil. Esta compatibilidade permite que este sistema automático seja muito útil no auxílio das tarefas de segmentação e anotação manual do vídeo.

Ao analisar globalmente o trabalho desenvolvido é importante ter em conta a restrição imposta às metodologias a adotar na segmentação por *shots* e por histórias. A obrigatoriedade do funcionamento em tempo real condicionou as abordagens a só poder ter acesso à informação presente e passada.

Para tal começou-se por recolher alguns vídeos por forma a criar um *corpus* de trabalho que fosse representativo dos principais programas noticiosos do horário nobre português.

Esse *corpus* foi dividido em dois conjuntos, um conjunto de treino para desenvolver e e otimizar os algoritmos e um conjunto de teste para avaliar o desempenho dos algoritmos e compará-lo com o estado da arte (denominados por *corpus* de treino e *corpus* de teste, respetivamente).

A recolha do corpus permitiu definir uma estratégia de segmentação que passou

por segmentar o vídeo ao nível do *shot*, proceder à seleção de *keyframes* e aplicar um algoritmo de deteção de pivô às *keyframes* selecionadas.

Por fim, foram efetuados os testes finais sobre o *corpus* de teste onde se obtiveram resultados bastante satisfatórios (quando comparados com o estado da arte) para o algoritmo de segmentação por *shots*, tendo em conta que este processa o vídeo à medida que é capturado.

No que se refere aos resultados obtidos com o algoritmo de segmentação por *shots*, e comparando-os com os resultados obtidos em 2005 no TRECVID (tabela 6.1) podemos verificar que este trabalho apresentou melhores resultados.

Tabela 6.1: Resultados da cobertura comparativamente ao TRECVID 2005 [32].

Tipo	TRECVID	Trabalho
Transições Abruptas	93.6 %	94.7 %
Transições Suaves	78.8 %	89.6 %

Observando a tabela apresentada, e de acordo com a pesquisa bibliográfica efetuada, pode-se ainda concluir que as transições suaves são geralmente mais difíceis de detetar, pelo que possuem um menor valor de *recall*.

No que toca ao algoritmo de segmentação por histórias também se obteve uma melhoria de resultados quando comparando com o TRECVID de 2004 (com uma melhoria de cerca de 10% da métrica F_1). Estes resultados revelam que esta técnica de deteção de pivô é bastante promissora em termos de poder vir a proporcionar uma ferramenta automática de segmentação temporal de programas noticiosos [33].

Relativamente ao desempenho global dos algoritmos há ainda que referir que estes foram capazes de processar um programa noticioso em apenas 40% da duração real do programa. Neste aspeto as três componentes responsáveis pela duração foram a deteção de contornos, o espessamento de contornos e a deteção facial.

6.1 Desenvolvimentos futuros

Do ponto de vista dos possíveis desenvolvimentos futuros uma das técnicas que se tem revelado mais promissora é a utilização de *General-Purpose Computing on Graphics Processing Units* (GPGPU) para a análise e processamento de vídeo. Este facto permite fazer cálculos de forma paralela com muito pouco *overhead* o que se adequa às técnicas de deteção e espessamento de contornos utilizadas.

A eficiência do ponto de vista computacional do algoritmo de segmentação por histórias poderá também ser melhorada reduzindo o tempo dedicado à deteção facial excluindo certas zonas da *frame* onde um pivô não estaria.

Em termos de resultados considera-se que o algoritmo de segmentação por histórias apresenta ainda oportunidades de otimização. Um dos aspetos que pode ser melhorado é a comparação de histogramas utilizando algoritmos de aprendizagem automática ou aproximações ao modelo DeltaE2000. Alternativamente à comparação por cor também é possível efetuar reconhecimento facial (desde que esta seja computacionalmente eficiente) identificando se a *keyframe* possui a mesma pessoa da *Template-pivô*.

Relativamente ao algoritmo de segmentação por *shots* é possível melhorar o seu desempenho ao nível dos falsos positivos. Nomeadamente a utilização de um algoritmo que o complemente por forma a minimizar falsos positivos provenientes de *flashes* fotográficos ou o aparecimento (ou desaparecimento) de legendas.

Bibliografia

[1] "Flickr." http://www.flickr.com/. [Online; Acedido a 03-Junho-2013].

[2] G. Ahanger and T. D. C. Little, "A survey of technologies for parsing and indexing digital video," *Journal of visual Communication and image representation*, vol. 7, pp. 28–43, 1996.

[3] G. Boccignone, A. Chianese, V. Moscato, and A. Picariello, "Foveated shot detection for video segmentation," *IEEE Transaction on Circuit and Systems For Video Technology*, vol. 15, pp. 365–377, 2005.

[4] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," pp. 511–518, 2001.

[5] NASA, "Anthropometry and biomechanics." http://msis.jsc.nasa.gov/sections/section03.htm. [Online; Acedido a 15-Junho-2012].

[6] M. Bertini, A. Bimbo, and P. Pala, "Indexing for reuse of tv news shots," *Pattern Recognition*, vol. 35, no. 73, pp. 581–591, 2002.

[7] C. Petersohn, *Temporal Video Segmentation*. Vogt Verlag, 2010.

[8] J. Yuan, H. Wang, L. Xiao, W. Zheng, J. Li, F. Lin, and B. Zhang, "A formal study of shot boundary detection," pp. 168–186, 2007.

[9] I. Koprinska and S. Carrato, "Temporal video segmentation: A survey," pp. 477–500, 2001.

[10] W. Fernando, C. Canagarajah, and D. Bull, "Wipe scene change detection in video sequences," *International Conference on Image Processing*, pp. 294–298, 1999.

[11] H. Sundaram and S.-F. Chang, "Determining computable scenes in films and their. structures using audio-visual memory models," pp. 95–104, 1999.

[12] D.-D. Le, S. Satoh, T. D. Ngo, and D. A. Duong, "A text segmentation based approach to video shot boundary detection," *IEEE 10th Workshop on Multimedia Signal Processing*, pp. 702 – 706, 2008.

[13] W. Abd-Almageed, "Online, simultaneous shot boundary detection and key frame extraction for sports videos using rank tracing," *IEEE 15th International Conference on Image Processing*, pp. 3200 – 3203, 2008.

[14] H. Zhang, A. Kankanhalli, and S. Smoliar, "Automatic partitioning of full-motion video," *Multimedia Systems*, vol. 1, pp. 10–28, Jan. 1993.

[15] W. Kim and J. Kim, "An adaptive shot change detection algorithm using an average of absolute difference histogram within extension sliding window," *IEEE 13th International Symposium on Consumer Electronics*, pp. 394–397, 2009.

[16] H. Jiang, A. Helal, A. Elmagarmid, and A. Joshi, "Scene change detection for video database management systems - a survey," 1996.

[17] T. Danisman and A. Alpkocak, "Dokuz eylül university video shot boundary detection at trecvid 2006," 2006.

[18] A. Dailianas, R. Allen, and M. Bellcore, "Comparison of automatic video segmentation algorithms," pp. 2–16, 1995.

[19] H. Fang, J. Jiang, and Y. Feng, "A fuzzy logic approach for detection of video shot boundaries," *Pattern Recognition*, pp. 2092–2100, 2006.

[20] R. Zabih, J. Miller, and K. Mai, "A feature based algorithm for detecting and classifying scene breaks," *ACM Multimedia*, pp. 189–200, 1985.

[21] C. Cotsaces, N. Nikolaidis, and I. Pitas, "Video shot boundary detection and condensed representation: A review," *Signal Processing Magazine, IEEE (Volume:23 , Issue: 2)*, pp. 28–37, 2006.

[22] D. Lelescu and D. Schonfeld, "Statistical sequential analysis for real time video scene change detection on compressed multimedia bitstream," *IEEE Transacitions on Multimedia*, 2003.

[23] Y. Zhai, A. Yilmaz, and M. Shah, "Story segmentation in news videos using visual and text cues," pp. 92–102, 2005.

[24] A. Goyal, P. Punitha, F. Hopfgartner, and J. M. Jose, "Split and merge based story segmentation in news videos," pp. 766–770, 2009.

[25] H. Misra, F. Hopfgartner, A. Goyal, P. Punitha, and J. M. Jose, "Tv news story segmentation based on semantic coherence and content similarity," pp. 347–357, 2010.

[26] L. Xie, L. Zheng, Z. Liu, and Y. Zhang, "Laplacian eigenmaps for automatic story segmentation of broadcast news," *Trans. Audio, Speech and Lang. Proc.*, vol. 20, pp. 276–289, Jan. 2012.

[27] A. G. Hauptmann and M. J. Witbrock, "Story segmentation and detection of commercials in broadcast news video," pp. 168–179, 1998.

[28] G.-J. Poulisse, M.-F. Moens, T. Dekens, and K. Deschacht, "News story segmentation in multiple modalities," *Multimedia Tools Appl.*, vol. 48, pp. 3–22, May 2010.

[29] M. Kipp, "Anvil-software." `http://anvil-software.de/`, 2008. [Online; Acedido a 16-Abril-2012].

[30] L. Shao and L. Ji, "Motion histogram analysis based key frame extraction for human action/activity representation," *Canadian Conference on Computer and Robot Vision*, pp. 88–92, 2009.

[31] M. Chatzigiorgaki and A. N. Skodras, "Real-time keyframe extraction towards video content identification," pp. 934–939, 2009.

[32] A. Smeaton, P. Over, and A. Doherty, "Video shot boundary detection: Seven years of trecvid activity," *Computer Vision and Image Understanding*, pp. 411 – 418, 2009.

[33] W. Hsu, L. Kennedy, S.-F. Chang, M. Franz, J. Smith, and G. Iyengar, "Adaptive feature discovery for trecvid broadcast news video story segmentation." `http://www-nlpir.nist.gov/projects/tvpubs/tvpapers04/ibm.story.slides.pdf`. [Online; Acedido a 12-Dezembro-2012].

Apêndice A

Algoritmo de avaliação

Por forma a avaliar o desempenho dos algoritmos de segmentação, é necessário confrontar os dados da segmentação manual com os dados que resultam da execução dos algoritmos. Para o efeito desenvolveu-se um *script* que, com base nestes dados, vai gerar uma tabela com quatro colunas (tabela A.1).

Estes dados irão ser utilizados na criação de uma tabela com 4 colunas. Esta tabela, para além do número da *frame*, inclui também a hipótese (os resultados da segmentação automática), a referência (os resultados da segmentação manual) e um descritivo do tipo de transição.

Na tabela A.1 podemos observar um Tp (nas *frames* 1001 e 1002) ,um Fp (*frame* 1005) e um Fn (nas *frames* 1008 e 1009).

Com o número da *frame*, pode-se fazer uma análise detalhada e determinar, utilizando o vídeo e a curva característica, a causa de um determinado Fp ou Fn.

Note ainda que durante a fase de segmentação manual uma transição dura sempre mais do que uma *frame*. Este efeito é particularmente visível no caso das transições suaves onde a zona de transição pode atingir dezenas de *frames*.

A coluna de descritivo possui ainda um identificador único para cada transição manual, por forma a filtrar situações como a apresentada na *frame* 1002. Esta *frame* deverá ser tratada em conjunto com a *frame* 1001 (visto que pertencem à

Tabela A.1: Tabela que associa a segmentação automática com a segmentação manual.

Número da Frame	Hipotese	Referência	Descritivo
...	...	...	...
1000	0	0	-
1001	1	1	Cut-3
1002	0	1	Cut-3
1003	0	0	-
1004	0	0	-
1005	1	0	-
1006	0	0	-
1007	0	0	-
1008	0	1	Cut-4
1009	0	1	Cut-4
...	...	...	...

mesma transição) e este descritivo permite fazer essa associação.

Este descritivo permite ainda fazer uma análise por conjunto. Isto é, uma análise em que se calcula a *precision* e *recall* para todas as transições de um determinado tipo (por exemplo *cuts* ou *dissolves*).

Devido há existencia de *subsampling* foi também dada uma tolerância de 3 frames para a detecção. Por exemplo, para o caso da transição da *frame* 1001, a transição seria considerada como uma deteção caso o algoritmo encontrasse um evento entre as *frames* 998 e 1005.