Pan, Zoom, Scan – Time-coherent, Trained Automatic Video Cropping

Thomas Deselaers, Philippe Dreuw, and Hermann Ney
Human Language Technology and Pattern Recognition Group
RWTH Aachen University, Aachen, Germany
`<lastname>@cs.rwth-aachen.de`

Abstract

We present a method to fully automatically fit videos in 16:9 format on 4:3 screens and vice versa. It can be applied to arbitrary aspect ratios and can be used to make videos suitable for mobile viewing devices with small and possibly uncommonly sized displays. The cropping sequence is optimised over time to create smooth transitions and thus leads to an excellent viewing experience. Current televisions have simple and often disturbing methods which either show the centre region of the image, distort the image, or pad it with black borders. The technique presented here can fully automatically find the "right" viewing area for each image in a video sequence. It works in real-time with only very little time-shift. We employ different low-level features and a log-linear model to learn how to find the right area. The method is able to automatically decide whether padding with black borders is necessary or whether all relevant image areas fit on screen by cropping the image. Evaluation is done on ten videos from five different types of content and the baseline methods are clearly outperformed.

1. Introduction

For decades, televisions and movies have been 1.33 times as wide as they were high, referred to as 4:3 or 1.33:1 television format. Today, the growing availability of wide-screen televisions and wide-screen movies often leads to unsightly video presentation: 4:3 movie presentation on 16:9 displays is disturbing in the same way as 16:9 (or wider) content is disturbing on conventional 4:3 displays.

Although increasingly many televisions have displays in 16:9 (1.78:1) format [4], still many old TV sets with 4:3 displays are in use and many of the increasingly popular mobile multimedia devices have uncommon aspect ratios. Due to the size of their displays, padding the images and effectively using only a small portion of the display is unacceptable. For example, the Apple iPhone$^{\text{TM}}$ has a screen diagonal of 3.5 inches and a resolution of 480×320 pixels, *i.e.* an aspect ratio of 3:2 when viewed in landscape mode, which makes approx. 29% of the screen unused when viewing videos in 16:9 format with black padding. Note that it is also possible to use it in portrait mode which leads to a display which is higher than wide.

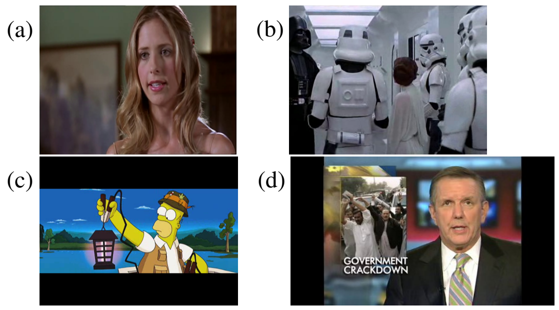

Figure 1: Examples of conventional handling of different aspect ratios for video material and display device. (a) squeezed, (b) cropped, (c) letterboxed, (d) pillarboxed.

Preliminaries. Regardless of the chosen display, the original aspect ratio of movies usually varies between 1.33:1 and 2.40:1. The common solutions to viewing content in an aspect ratio different from that of the display device are either to squeeze the image onto the screen (Fig. 1 (a)), to crop the image (Fig. 1 (b)) such that only a central area is visible, or to horizontally (Fig. 1 (c)) or vertically (Fig. 1 (d)) pad the images with black borders (letterboxing/pillarboxing).

Squeezing the images leads to distorted images which can be disturbing and is unacceptable in most cases (cp. Fig. 1 (a)). Padding the images leaves large portions of the display unused and thus makes the image smaller which is particularly annoying on mobile devices (cp. Fig. 1 (c), (d)). Cropping the image may lead to loss of important details of a movie, because up to 50% of the original picture can be lost (cp. Fig. 1 (b), where half of a main actor is lost).

Available Approaches. Today, most TV sets have several modes to fill the complete screen with videos of different aspect ratios. Some offer compromises between stretching and zooming to fill the screen, *e.g.* stretching the border areas of the images more than the centre part, which makes people in the centre of the screen appear correctly, others crop only a little so that they do not have to stretch as much. They all have in common that either potentially

978-1-4244-2243-2/08/$25.00 ©2008 IEEE

important image content is lost, or that an anamorphic image distortion occurs. As both is unacceptable in most cases and for most end-users, some DVDs are produced with a sequence of manually annotated cropping areas to ensure that the relevant areas of the image are always included in the cropped sequence. The process of finding the *right* cropping sequence is referred to as "*panning and scanning*". An operator manually selects the parts of the original composition that are significant (*i.e.* scanning). Once, the action shifts to a new position in the frame, the operator moves the region-of-interest, effectively following the action, creating the effect of a so called pan shot. This method allows for using the full screen resolution, but (a) requires costly manual annotation, which can be done for one or two different aspect ratios but not for arbitrary aspect ratios, and (b) due to the major reduction of cinema movies in size, the dynamic feeling and important image regions can be lost and thus effectively, the arrangement of a carefully composed scene can be destroyed.

On other types of video footage, *e.g.* live-sports or news, the scenes are not always composed this carefully, but the "*important objects*" are kept in the central part of the images. Here, problems arise, when top and bottom parts are cropped to reduce a 4:3 movie to 16:9 format and subtitles or news tickers are superimposed on the image.

In this work, we present a method that finds a pan, scan, and zoom sequence for a given video sequence such that no important content is lost, independent from the position in the frames, and a smooth transition between succeeding cropping positions is created. To achieve a panning and scanning sequence with as little loss of important image details as possible, we allow for slightly padding the image with black borders, *i.e.* zoom out of the image, for partial sequences. Smoothness of the resulting cropping sequence is obtained by explicitly optimising over time. The whole method is capable to run in real-time with only little time-shift. Starting from a general, feature-driven setup we develop a system that can find the best viewing pane for each frame in a video using a task-dependent and automatically learnt scoring method. To learn the parameters, only few frames from videos of the same domain as the target video have to be hand-labelled. Additionally, it is possible to learn a model that performs well on all types of video content.

Related work. The problem of automatic pan and scan for videos was addressed in [10] and [13] and for still images in [9]. The authors find a region of interest and create a warping function, *i.e.* apply a piecewise linear scaling function to the images, which leaves the region of interest unchanged and warps the rest of the image. The video retargeting is smoothened by adding constraints for consecutive frames. Contrary to the approach presented here, in [13] there is no explicit optimisation over time, no zooming out, no learning, and no quantitative evaluation.

A novel method for image retargeting has recently been presented in [1]. Avidan and Shamir propose a method that allows for content-aware resizing of images, *i.e.* keeping the complete content of an image by removing '*irrelevant*' regions. This method resizes images by bringing the content closer together. We quantitatively compare our approach with this method, which allows some interesting insights.

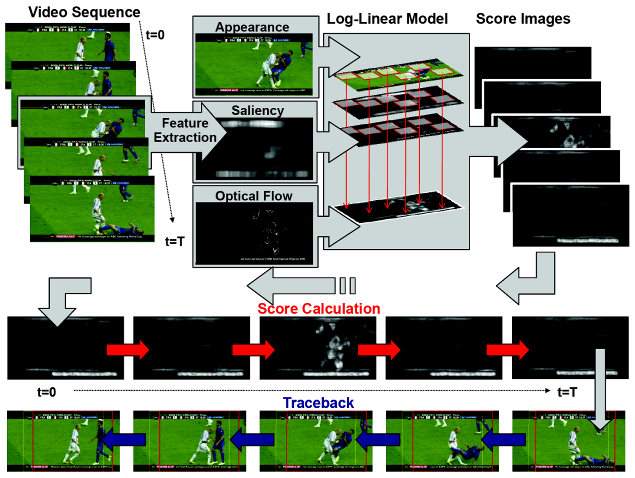

Figure 2: Overview of the whole system. First (top part) from each frame of the video sequence, features accounting for appearance, saliency, and motion are extracted, a log-linear model is applied to each pixel of the image potentially using these three cues, resulting in a score-image per frame. In the second phase (bottom part), dynamic programming is applied to find a sequence of viewing panes for the video frames. In the forward step, the scores for all potential areas are computed and in the backward/traceback step, the best viewing sequence is determined.

A similar problem is addressed by the drivers of many webcams and video chat applications: the webcam has a wide angle lens to make sure that the user's face is captured, then the software detects the face in the image and recenters the display window around the face. There, face detectors and possibly motion are used to find the viewing window. More advanced methods allow for segmenting the webcam user from arbitrary background in monocular video [3, 14] (such as considered here) and in binocular video [8].

In [5], a similar algorithm as the one proposed here for optimisation is used for tracking of hands and faces for sign language recognition.

Related problems include viewing large images on mobile devices where different solutions are presented in [2] and [11]. It is proposed to detect the most important region and automatically move the view-port over the image.

2. Overview

The system is built from two basic components: (1) feature extraction and combination, (2) panning, scanning, and zooming. An overview of the system is given in Fig. 2.

In the first component, the images are analysed to find relevant regions. To determine whether a region is relevant we employ three different strategies, namely visual saliency of still images, optical flow to capture movements, and a trained scoring algorithm which can use the three cues accounting for saliency, motion, and the appearance of the image to determine relevance for each position of an image. The parameters for this scoring are trained once for each type of video content, and can then be used for arbitrary sequences of the same or similar content-types. The algorithm can also be applied to video content types for which no training has been performed at all. Then, the log-linear scoring is replaced by a weighted (possibly controllable by the user) sum of saliency and optical flow.

In the second component, the output of the first component is analysed to find a sequence of cropping positions with correct aspect ratio for the display device, shows as many relevant image parts as possible, is smooth to give a nice viewing experience, and uses as much of the available display, *i.e.* applies as little black padding, as possible.

The cropping sequence is potentially optimised over the whole video using dynamic programming. Ideally it is applied shot-wise, *i.e.* shot boundary detection is applied and then the best cropping sequence for each shot is determined. Using early tracebacks in the dynamic programming, the cropping sequence can be found with little time-shift, *i.e.* 30 frames (approx. one second) of time-shift leads to hardly any decrease in performance or smoothness of the method but allows a near-live viewing. The system itself, which is an unoptimised implementation in C++ using OpenCV, is able to process videos in near real-time.

In general the method can be applied to any type of video footage. Only a little time-shift is required to be able to apply the method with sufficiently good results.

3. Task Definition

To evaluate and test our system, we defined a set of 5 different types of video content representing a wide variety to show that the system generalises over different types of video content: TV series (*Buffy, the vampire slayer*), cinema movies (*Star Wars*), Sport (*football world cup 2006*), Cartoon (*Simpsons movie*), and News shows. For each of these, we have two sequences of at least 30 seconds. To be able to evaluate the performance of the method we randomly selected 10 frames from each video and labelled some areas of the images as important and unimportant, respectively. An example image from each type of video content together with the corresponding relevance labellings is shown in Fig. 3 (a) and (b). To keep training and testing data disjoint, we use the respective other video of the same type. In particular the selected sequences are fully disjoint and from different parts of the movies. Note that we chose five different types of video content to show that the presented approach in principle works for arbitrary content type.

TV series: Buffy. The Buffy sequences are two, 30-second long, non-overlapping parts of the second episode from the fifth season from "Buffy the Vampire Slayer". The images are of size 640 $\times$ 340 (*i.e.* 1.88:1 original aspect ratio), the target aspect ratio is 4:3.

News. One sequence from ABC World News and one sequence from BBC World News were collected, lasting 30 seconds each. ABC World News has an aspect ratio of 4:3 (image size 320$\times$240) and BBC World News has an aspect ratio of 16:9 (416$\times$232 pixels). The target aspect ratio is 16:9 for the first and 4:3 for the second.

Cartoon: Simpsons Movie. Two different, disjunct parts of two trailers of the Simpsons movie were selected. The images are of size 640 $\times$ 272 (*i.e.* 2.35:1 original aspect ratio), the target aspect ratio is 4:3.

Sport: Football World Cup 2006. One minute from the world cup game Iran vs. Mexico and one minute from the world cup final France vs. Italy were selected. The images are of size 640 $\times$ 340 (*i.e.* 1.88:1 original aspect ratio), the target aspect ratio is 4:3.

Cinema: Star Wars IV: A New Hope Two 30-second long, non-overlapping parts of Star Wars IV were randomly selected. The images are of size 640$\times$320 (*i.e.* 2.00:1 original aspect ratio), the target aspect ratio is 4:3.

Creation of Ground Truth. To create the ground truth, which is required for training and for evaluating the system, a user is presented with ten frames taken from each video and marks areas as important (red) and unimportant (blue) by painting into the image. The user is not required to label each pixel, unlabelled pixels are not considered at all and the labelling can be very coarse (Fig. 3). This way, frames can be annotated quickly. A user presented with the annotation system for the first time was able to annotate an image within few seconds.

Experimental Protocol. For the evaluation, we give the recall (percentage of important pixels displayed) and the percentage of unused pixels on the display (*i.e.* if the image is padded with a black border). Ideally, all important pixels are shown without having to matte the image with black borders.

4. Features

Feature-based relevance of image regions can be determined by saliency, motion is captured by optical flow. Saliency, optical flow, and appearance can furthermore be fused using a log-linear model to determine a probability of relevance. Specialised detectors could be used for some types of content, *e.g.* a face detector, such as the one presented in [12] to mark faces as relevant. Note, that simple difference images or background subtraction is not sufficient to determine the important areas in a video sequence due to possible camera movements, zooming, and changing backgrounds. To allow the system to learn which parts of the images are relevant, depending on saliency, optical

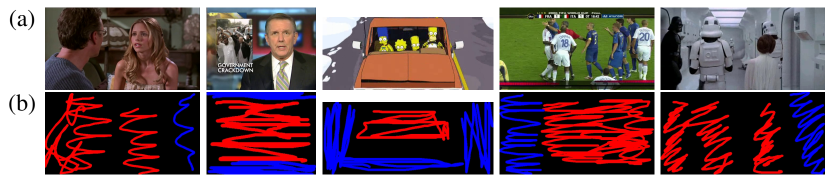
Figure 3: Example images for the 5 different types of video (top) and their corresponding relevance annotation (bottom). Red annotation denotes important areas and blue annotation denotes unimportant areas. From left to right: Buffy, News, Simpsons, Football, and Star Wars.

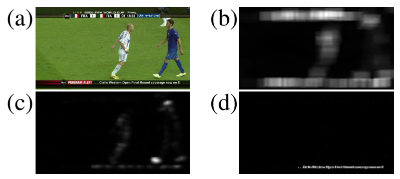
Figure 4: Features: (a) image, (b) saliency, (c) video saliency, and (d) optical flow.

flow, and appearance, we propose to train a simple log-linear model effectively fusing these three cues to yield one relevance-score per pixel. If desired, it is possible to easily add a face-cue (obtained from a face detector) to the cues considered here.

4.1. Saliency

Computational modelling of visual saliency is a challenge. In [7] a simple method for visual saliency detection is presented. The log-spectrum of a given input image is analysed to extract the residual of an image in the spectral domain. The method was shown to perform well on both natural images and artificial, psychological patterns and performs fast since it mainly requires to calculate two two-dimensional Fourier transforms. Given an image X, the saliency map $\mathcal{S}(X)$ is obtained as

$$\mathcal{S}(X) = \mathcal{F}^{-1}\big(\exp(\mathcal{R}(X) + \mathcal{P}(X))\big)^2 * g_1 \text{ with} \quad (1)$$

$$\mathcal{R}(X) = \log(||\mathcal{F}(X)||) - \log(||\mathcal{F}(X)||) * g_2 \quad (2)$$

$$\mathcal{P}(X) = \Im\left(\mathcal{F}(X)\right), \quad (3)$$

where $\mathcal{R}(X)$ is the spectral residual, $\mathcal{P}(X)$ is the phase of the Fourier transform $\mathcal{F}(X)$, *i.e.* the imaginary part of the Fourier transform, g_1 and g_2 are smoothing filters, and $\mathcal{F}$ and $\mathcal{F}^{-1}$ denote the Fourier transform and the inverse Fourier transform, respectively.

It is straightforward to extend the method to calculate the saliency on 3-dimensional data and thus it could be used to determine saliency in videos. An example saliency map and the corresponding *video saliency map* are shown in Fig. 4 (b) and (c). The video saliency was obtained for a 63-image sequence with 31 images before and 31 images after the frame of interest. In the video saliency, static but otherwise salient objects are not salient anymore. For our application a region is likely to be important if it is salient *or* if something is moving rather than if it is salient *and* it is moving. Therefore, we did not use video saliency but only frame-wise image saliency in the experiments.

4.2. Optical Flow

Optical flow describes the motion of objects in a scene and is a well understood way to capture motion in videos. The optical flow is a flow-field specifying the motion in x and y-direction for each pixel in an image X. The calculation of the optical flow is commonly based on the assumption that given a sufficiently high sampling frequency, the intensity of a particular point is constant. We use the OpenCV implementation of the method described in [6] which gives us a dense flow field in x-direction $\mathcal{X}(X)$ and in y-direction $\mathcal{Y}(X)$. To capture motion, we use the magnitude of the flow vectors $\mathcal{O}(X) = \sqrt{\mathcal{X}(X) + \mathcal{Y}(X)}$.

An example optical flow magnitude $\mathcal{O}(X)$ is shown in Fig. 4 (d). The expectation is that optical flow is able to capture where in the image motion occurs as opposed to the saliency which is expected to find interesting regions in static images. In particular, optical flow is expected to help when subtitles or news tickers are superimposed on images. For example, the title line in a football game showing the current score of a game is not required to be visible always, but if information about a player exchange is superimposed in the bottom part, this information may be important.

5. Log-linear Scoring

One of the core components of the system is a log-linear model which can be parametrised task dependently. Given the saliency, the optical flow, and the appearance (RGB channels), for each position (x, y) in an image X we calculate a probability whether the position is important (denoted as 1, opposed to irrelevant, denoted as 0) as

$$p(1|X,x,y) = \frac{\exp\left(\alpha_1 + \sum_{i=0}^{I} \lambda_{1i} f(X,x,y,i)\right)}{\sum_{c\in\{1,0\}} \exp\left(\alpha_c + \sum_{i=0}^{I} \lambda_{ci} f(X,x,y,i)\right)}, \quad (4)$$

where $f(X, x, y, i)$ denotes the i-th feature extracted for position (x, y) in image X. λ_{ci} and α_c are the parameters of the log-linear model and are trained with gradient descent.

Feature functions $f(X, x, y, i)$. To allow the log-linear model to fuse the three cues, we use sub-windows from each cue as features. The window size was chosen to be 15×15 pixels, *i.e.* 225 features are extracted for saliency and optical flow and 675 features are extracted for appearance. In the experiments presented later, we show the impact of the different cues on the results and how fusing them helps to improve the results by combining their advantages. A relevance probability $p(1|X, x, y)$ is obtained by applying Eq. 4 after the features $f(X, x, y, i)$, which effectively are pixel values, saliency score, or optical flow are extracted. This procedure is schematically presented in Fig. 5.

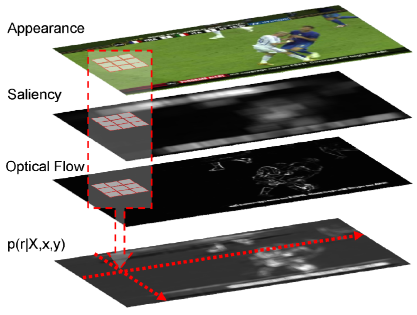

Figure 5: Calculation of $p(1|X, x, y)$ using different cues.

Training of the Parameters. To train the parameters λ_{ci} and α_c we extract the required features from all positions in the labelled images of the training sequences and use them as training data for class 1 if it was labelled red and for class 0 if it was labelled blue. Unlabelled, *i.e.* black, pixels are not considered for the training. For each video sequence, we extract the features for each annotated image and train one model per content-type. Note that we always use one sequence for training and the other for testing, resulting in a total of 10 training and 10 evaluations per setting to keep training and testing data disjoint.

6. Automatic Panning, Zooming, and Scanning - Finding the Best Cropping Sequence

Given the scores from the first step, in this section we describe how we find the optimal cropping sequence for a given video sequence to achieve a smooth and time-coherent impression. One possible way to find a cropping sequence is to detect the best scored area in each frame individually, leading to a non-smooth cropping sequence with large jumps between succeeding frames which is unacceptable for most viewers. Instead, we use dynamic programming, which, in the forward step accumulates scores for potentially hypothesised cropping sequences and in the backward step (traceback) determines the optimal one. By defining suitable penalty functions it is possible to define exactly which transitions are allowed between succeeding frames and thus preventing non-smooth cropping sequences. The optimisation over the sequence then finds the best smooth sequence. Using early traceback it is possible to use this method with only little time shift in an on-line manner.

The proposed algorithm prevents taking possibly wrong local decisions, because the decision making is done at the end of a (partial) sequence by tracing back the decisions to construct the optimal path w.r.t. to the given criterion. In the following, we describe the two steps of the algorithm. First, the algorithm is described for a fixed size pan and scan area and then extended toward variable pan, scan, and zoom sizes. An overview of this algorithm is given in the bottom part of Fig. 2.

Step 1: Accumulation of Scores. For a frame X_t at time step $t = 1, ..., T$, for each position (x, y) in X_t a local score $q_t(x, y, w, h)$ for a possible pan and scan area of size $w \times h$ pixels centred at (x, y) is calculated as the sum of the relevance-scores over this area. This can be calculated efficiently using integral images I_t [12]:

$$q_t(x,y,w,h) = I_t\left(x+\frac{w}{2}, y+\frac{h}{2}\right) - I_t\left(x+\frac{w}{2}, y-\frac{h}{2}\right) - I_t\left(x-\frac{w}{2}, y+\frac{h}{2}\right) + I_t\left(x-\frac{w}{2}, y-\frac{h}{2}\right) \quad (5)$$

The local score $q_t(x, y, w, h)$ thus gives the sum over the scores in a hypothesised pan and scan area. From these local scores, for each time step t a global score $Q_t(x, y)$ is calculated for each position. $Q_t(x, y)$ is the score for the best hypothesised cropping sequence ending at time t in position (x, y), the according best predecessor is stored in a table of back pointers $B_t(x, y)$, *i.e.* the position (x', y') which leads to the score $Q_{t-1}(x', y')$ in the preceding frame and is passed to come to (x, y) in the frame at time t. The stored predecessors are used in step 2 to trace back the optimal cropping sequence.

The recursive equation for this dynamic programming algorithm is defined as follows:

$$Q_t(x,y) = \max_{(x',y') \in R(x,y)} \left\{ (Q_{t-1}(x',y') - \mathcal{T}(x',y',x,y) \right\} + q_t(x,y,w,h) \quad (6)$$

$$B_t(x,y) = \arg\max_{(x',y') \in R(x,y)} \left\{ (Q_{t-1}(x',y') - \mathcal{T}(x',y',x,y) \right\} \quad (7)$$

where $R(x, y)$ is the set of possible predecessors of point (x, y) and $\mathcal{T}(x', y', x, y)$ is the transition-penalty from point (x', y') in the predecessor image to point (x, y) in the current image. The transition-penalty can be used to penalise large jumps between succeeding frames, by *e.g.* using the Euclidean distance between two succeeding points (x, y) and (x', y'): $\mathcal{T}(x', y', x, y) = \alpha \cdot \sqrt{(x - x')^2 + (y - y')^2}$ where α is a weighting parameter to control smoothness vs. flexibility.

Step 2: Tracing Back the Optimal Path. Given the accumulated scores $Q_t(x, y)$ over a sequence of frames for time steps $t = 1 \ldots T$, we begin from the last frame at time T to reconstruct the optimal path, which is a sequence of cropping positions. A full traceback starts from the last frame of the sequence at time step T using the cropping position $(x_T, y_T) = \arg\max_{x,y} Q_T(x, y)$. Then, the best tracking centre at time $T-1$ is looked up in the table of back pointers $(x_{T-1}, y_{T-1}) = B_T(x_T, y_T)$. This is iteratively repeated until a cropping position for each time step $t = T \ldots 1$ is determined.

In a full traceback, the decision for a single frame automatically depends on all preceding and succeeding decisions. Using early tracebacks over Δ frames (*e.g.* $\Delta = 30$), the decisions for each frame only depend on the frames which are considered in the same partial optimisation. The optimisation can be initialised with the outcomes from

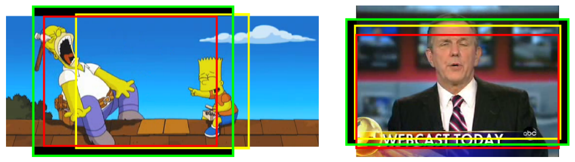

Figure 6: Two examples with crop marks where slight padding (left: letterboxing, right: pillarboxing) is necessary to avoid loss of important image regions: yellow, centre crop markings; red, automatic pan and scan; green, automatic pan, scan, and zoom.

a preceding decision which makes it possible to obtain smooth cropping sequences even if the path is not optimised over the whole sequence. If $\Delta = 0$ is chosen, completely local decisions are taken and a non-smooth cropping sequence is obtained.

Extensions. As described above, there are cases, where a simple pan and scan is not able to capture all relevant parts of the image, because the relevant area is larger than fits on the screen. In these cases, it is desired to slightly zoom out, and apply black mattes to the borders of the image. Examples for this are given in Fig. 6, where it can be observed that central cropping leads to loss of large parts of the relevant image regions, automatic pan and scan has a slightly better result where more of the relevant information is kept, and the automatically padded image, despite being padded only marginally, shows all relevant parts. Here, time-coherence is particularly important since otherwise the zooming in and out might lead to disturbing effects, but if the zooming is done slowly and smoothly the impression is similar to a smooth camera zoom. This is made possible by optimising over time which allows to start zooming out relatively long before and thus a smooth transition is possible.

To allow for automatically changing the size of the pan and scan area, we adapt our recursive equation, our score, and our penalty functions. To account for varying sizes of the boxes, we normalise the scores with the size of the box and jointly optimise w.r.t. to position and size:

$$q_t(x,y,w,h) = \max_{\substack{W \le w' \le \delta W \\ H \le h' \le \delta H}} \left\{ \frac{q_t(x,y,w',h')}{(w' \cdot h')} \right\} \tag{8}$$

where the sizes W and H correspond to the dimensions of the default pan and scan area, *i.e.* no padding is required to fit the cropped image of size $W \times H$ pixels onto a screen with the desired aspect ratio, and δ is maximally allowed factor to deviate from the default target aspect ratio.

Penalty Functions. In Eq. 6, $\mathcal{T}(x', y', x, y)$ is introduced as transition penalty to penalise non-smooth cropping sequences, to further improve the viewing experience, we present some additional penalty functions:

Centre Penalty: Commonly we expect the most important areas to be central. To penalise non-central pan and scan areas, the following penalty can be incorporated

$$\mathcal{C}(x,y) = \beta \cdot (|x - x_c| + |y - y_c|), \tag{9}$$

where (x_c, y_c) is the centre of the image.

Aspect Ratio Penalty: To penalise diverging from the correct aspect ratio, *i.e.* to avoid padding with black borders, we incorporate the following penalty function

$$\mathcal{A}(w,h) = \gamma \cdot |(w/h) - C|, \tag{10}$$

where C denotes the aspect ratio of the target display (C=1.78 for a 16:9 display, C=1.33 for a 4:3 display).

Zoom Penalty: Frequent zooming in and out, *i.e.* padding with black borders and removing these, can be visually disturbing, to penalise changing the cropping size between succeeding frames we incorporate the zooming penalty

$$\mathcal{Z}(w', h', w, h) = \delta \cdot (|w - w'| + |h - h'|), \tag{11}$$

where (w, h) and (w', h') denotes the size of the current and of the predecessor's cropping area

These penalty functions can be incorporated into Eq. 6 and 7 in the same way as the transition penalty $\mathcal{T}$. The parameters α, β, γ, and δ are scaling factors that can be used to tune the influence of the individual penalty functions.

7. Experiments

First, we present results which are obtained when only panning and scanning is done. We compare the results to cropping the centre part of the image, and to the content-aware image resizing (CAIR) method presented in [1][1]. Tab. 1 shows the percentage of relevant pixels (recall) displayed using the centre cropped image (first line), CAIR (second line), purely feature driven method presented here (second block) and the trained method (bottom block). Note that in these experiments, the cropped region always has the target aspect ratio and thus no black borders are attached to the image. It can be observed that the recall for the centre cropped image in general outperforms the CAIR method, although visually the results from the CAIR method are good. In particular for the Simpsons scenes, quantitatively CAIR performs badly but the resulting images are nice. Example images for CAIR are given in Fig. 7. It can be seen that it works very well for the Buffy and Simpsons images. In the football sequence the distorted lines are disturbing.

The recall values for the purely feature driven setups and for the different setups of the trained method are higher than the baseline methods. In particular it can be observed that the results improve when all cues are used jointly. Only for the first football sequence the recall is only at about 90% for the best setting. For all other sequences the good settings are in the range of 96-100%.

In further experiments, we evaluated early tracebacks to reduce the time-shift when viewing a video. We evaluated tracing back every 0, 3, 30, and 300 frames and observed that with a very short traceback of 0 (local search) or 3 frames, the viewing experience is significantly deteriorated due to frequently occurring jumps between the re-initialised parts of the video, but the recall values are slightly im-

[1]`http://sourceforge.net/projects/c-a-i-r/`

Table 1: Results from panning and scanning using different techniques compared to simple centre cropping and content-aware image resizing (CAIR). All results are given as median percentage of the relevant pixels shown. First block: centre cropped and CAIR, second block: using only visual features, bottom block: trained system, A: appearance cue, OF: optical flow, S: saliency.

method	Buffy		News		Simpsons		Football		Star Wars	
	1	2	1	2	1	2	1	2	1	2
cen.crop	93.0	87.7	97.2	96.0	98.3	83.8	93.1	92.7	79.0	95.6
CAIR	87.2	75.3	80.9	87.9	73.3	71.7	92.2	90.3	78.7	87.4
OF	100.0	93.4	86.4	95.5	99.8	98.0	92.1	100.0	99.6	100.0
S	100.0	95.4	99.9	98.4	98.0	93.0	90.1	100.0	98.9	100.0
OF+S	100.0	95.4	99.9	98.4	98.7	94.4	90.1	100.0	99.4	100.0
A	99.9	95.3	98.2	91.9	97.2	91.9	88.3	99.7	97.6	100.0
OF	100.0	88.4	86.4	95.5	98.3	97.7	93.0	99.9	99.6	99.8
S	100.0	93.9	98.8	99.6	99.1	92.3	87.0	99.9	99.4	100.0
S+OF	100.0	95.1	98.8	99.2	99.2	93.2	92.4	100.0	99.4	100.0
S+OF+A	100.0	97.3	99.7	96.4	98.2	94.4	88.3	99.7	98.1	100.0

proved, due to the higher flexibility in the search. With 30 frames (slightly more than one second time-shift) these jumps disappeared nearly completely and with 300 frames (slightly more than 10 seconds time-shift), we could not distinguish it from a video with full traceback. The performance for the methods is hardly changed, thus we performed all further experiments with full tracebacks. Ideally, the optimisation would be performed after shot-boundary detection on each shot individually.

Next, we evaluate how the method performs when it is allowed to change the size of cropping area, *i.e.* if it is allowed to pad the image with black borders, to keep more image content. Ideally, the shown portion of the relevant image parts is maximised while the black padding is minimised. Tab. 2 gives results for these experiments, the first block gives results for central cropping of differently padded images, no padding, 10% wider than central padding, 30% wider than central padding, and full padding, *i.e.* no content is lost. The recall values for the padded runs are clearly improved but obviously the unused area of the screen is extended. The second, third, and fourth blocks give results from the experiments with the newly proposed

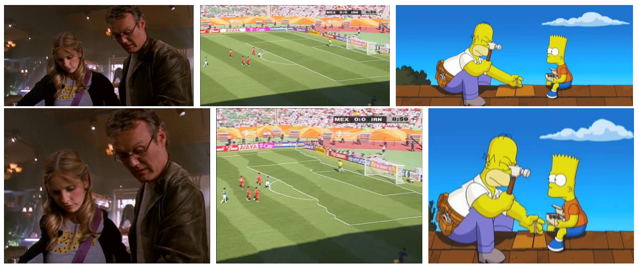
Figure 7: Example images for content-aware image resizing for the (from left to right) Buffy, Football, and Simpsons sequences (top: original image, bottom: resized).

method where it is allowed to change the aspect ratio by 30% maximal. The purely feature driven setups (Tab. 2, second block) show greatly improved recall values and relatively small values for the unused area which is clearly preferred over the strongly padded baseline runs. The trained setups (Tab. 2, third block) show even stronger improvements and only very small unused areas. The bottom block gives results when training is done on *all* first sequences and testing on *all* second sequences and vice versa. That is, the results in the bottom block are fully content-type independent. Note that for each of these runs the padding with black borders does *not* occur for the whole sequence (which is also possible) but only happens during short periods of time. It is smoothly added when required and removed when not required.

Comparing the average recall and black values, the centrally cropped run with 30% padding has an average recall of 98.6% and a high unused area of 23.0%. Our method (purely feature driven: row OF+S) obtains an average recall of 97.7% and only 2.9% of the screen is unused. The trained method using all three cues, has an average recall of 98.7% and 97.6% for the content-type dependent and independent setups, respectively. The average unused area for these runs is 2.2% and 2.5%, respectively. The content-type independent run is only slightly worse than the content-type dependent ones, which shows that the method generalises well and can be applied for arbitrary video sequences.

We performed various experiments with the different penalties and found that the centre penalty does not have a beneficial effect but rather limits the method. The zoom penalty, the aspect ratio penalty, and the transition penalty, however lead to visually much more appealing sequences due to reduced jitter and less zooming in and out. Although, with zooming penalty the average black area on the screen is slightly increased by 2-5%, the viewing experience is clearly improved.

To completely evaluate the advantages and disadvantages of the method presented, it may be necessary to perform experiments with users assessing the perceived video quality on different screens. However, the presented quantitative results indicate a large potential for improvements in perceived quality, which was also subjectively observed by looking at the results.

Runtime. The proposed method works in near real-time on all of the videos used and faster than real-time on the smaller news videos. Using only saliency or optical flow without training, the system processes 75 and 40 frames per second (fps), respectively. In the trained setups, it processes 15, 12, and 12 fps when using saliency, optical flow, or appearance, respectively. Appearance is not faster than the other two, due to the higher number of features(RGB channels). When using all three cues simultaneously, the system processes 8 frames per second. With an additional engineering effort and code cleanup, we assume that it is possible to

Table 2: Results from panning, scanning, and zooming. The columns of the table denote recall (cp. Tab. 1) and percentage of black/unused pixels on screen. The first block gives the results of different centrally cropped setups, with 0%, 10%, 30% divergence from the target aspect ratio and fully padded (*i.e.* nothing is lost) images. The second block gives results for the purely feature driven setups, the third block gives the results for the different trained setups, and the bottom block is obtained with content-type-independent training.

method	Buffy				News				Simpsons				Football				Star Wars			
	1		2		1		2		1		2		1		2		1		2	
cent. crop	93.0	0.0	87.7	0.0	97.2	0.3	96.0	0.0	98.3	0.0	83.8	0.0	93.1	0.0	92.7	0.0	79.0	0.0	95.6	0.0
cent.crop 10%	96.0	9.1	92.0	9.1	98.7	9.1	98.8	9.0	99.6	9.0	88.0	9.0	95.0	9.1	95.2	9.1	84.9	9.1	98.1	9.1
cent.crop 30%	99.2	23.1	99.0	23.1	100.0	23.1	100.0	22.9	100.0	22.9	94.7	22.9	99.2	23.1	98.8	23.1	95.3	23.1	100.0	23.1
fully padded	100.0	29.2	100.0	29.2	100.0	24.9	100.0	25.6	100.0	43.3	100.0	43.3	100.0	29.2	100.0	29.2	100.0	33.3	100.0	33.3
OF	100.0	0.0	91.4	2.0	86.4	0.9	97.5	3.7	100.0	7.4	98.0	5.5	91.7	0.3	100.0	3.6	97.9	1.2	100.0	5.2
S	100.0	2.1	88.9	2.9	99.9	1.5	98.4	3.1	99.5	9.9	95.9	6.4	90.1	0.0	100.0	0.0	99.5	1.8	100.0	1.2
OF+S	100.0	1.8	89.2	2.9	99.9	0.0	98.4	3.0	99.4	6.6	96.9	6.4	93.5	4.4	100.0	0.7	99.6	1.9	100.0	1.4
A	100.0	0.0	95.5	1.3	99.3	2.0	99.8	3.4	97.2	2.6	94.6	5.8	88.8	4.4	99.8	1.0	97.6	5.7	100.0	1.2
OF	100.0	0.1	99.4	11.6	89.6	0.3	95.5	3.1	99.8	3.8	97.7	3.7	93.3	0.1	99.9	2.8	98.3	1.1	100.0	4.9
S	100.0	1.7	95.5	1.4	99.7	0.9	99.6	1.7	99.5	9.7	97.7	6.6	92.3	3.5	99.9	0.0	99.7	5.3	100.0	1.2
S+OF	100.0	1.7	95.5	0.9	99.9	1.3	99.6	2.7	99.4	9.7	97.7	6.6	91.8	0.6	99.9	0.0	99.7	5.3	100.0	1.2
S+OF+A	**100.0**	**0.2**	**97.1**	**0.2**	**100.0**	**0.0**	**99.0**	**2.6**	**98.9**	**2.9**	**98.6**	**6.0**	**94.8**	**0.3**	**99.9**	**1.0**	**99.1**	**5.6**	**100.0**	**3.6**
S+OF+A (all)	100.0	0.0	92.8	0.3	100.0	4.5	99.8	2.1	95.7	5.8	97.7	2.3	92.0	1.2	99.4	0.1	98.1	5.6	100.0	3.6

use all setups in real-time (measurements were performed single-threaded on an Intel Core 2 Duo 2.4GHz processor).

8. Conclusion

We presented a method that allows for time-coherent automatic panning, scanning, and zooming to present video sequences on displays of different aspect ratios. In a first step, the method uses different visual cues which are fused using a log-linear model to obtain relevance scores. In a second step, dynamic programming is applied to find the optimal cropping path for a given video sequence. The method is evaluated on five different types of video content and is shown to be robust w.r.t. superimposed subtitles and camera motion. The method also allows to dynamically choose whether padding with black borders is necessary to show all important image parts, while keeping the amount of padding as low as possible. Different setups are presented, starting from a simple, low-level feature-based setup to a trained system that makes use of appearance, saliency and motion cues. It is shown that the method can be trained content-type dependent and that performance hardly decreases if performed content-type independent.

Acknowledgements. This work was partially supported by the DFG (German Research Foundation) under contract NE/572-6. We would like to thank Joseph Auman for providing his implementation of the content-aware image resizing method.

References

[1] S. Avidan and A. Shamir. Seam carving for content-aware image resizing. *ACM SIGGRAPH 2007*, 26(3), 2007.

[2] L. Chen, X. Xie, X. Fan, W. Ma, H. Zhang, and H. Zhou. A visual attention model for adapting images on small displays. *Multimedia Systems*, 9(4):353–364, 2003.

[3] A. Criminisi, G. Cross, A. Blake, and V. Kolmogorov. Bilayer segmentation of live video. In *CVPR 2006*, New York, NY, USA, June 2006.

[4] C. Daskalakis, B. Gessler, and N. Labourdette. Programme production and the 16:9 action plan: An assessment. *Int. Broadcasting Convention 1995*, pages 366–371, Sept. 1995.

[5] P. Dreuw, T. Deselaers, D. Rybach, D. Keysers, and H. Ney. Tracking Using Dynamic Programming for Appearance-Based Sign Language Recognition. In *FG 2006*, pages 293-298, Southampton, UK, April 2000.

[6] B. K. P. Horn and B. G. Schunck. Determining optical flow. *Artificial Intelligence*, (17):185–203, 1981.

[7] X. Hou and L. Zhang. General purpose object detection: A spectral residual approach. In *CVPR 2007*, Minneapolis, MN, USA, Jun 2007.

[8] V. Kolmogorov, A. Criminisi, A. Blake, G. Cross, and C. Rother. Probabilistic fusion of stereo with color and contrast for bi-layer segmentation. *IEEE Trans. PAMI*, 28(9):1480–1492, Sept. 2006.

[9] F. Liu and M. Gleicher. Automatic image retargeting with fisheye-view warping. In *ACM UIST*, pages 153–162, Seattle, WA, USA, Oct. 2005.

[10] F. Liu and M. Gleicher. Video retargeting: Automatic pan and scan. In *ACM Multimedia*, pages 241–250, Santa Barbara, CA, USA, Oct. 2006.

[11] H. Liu, X. Xie, W. Ma, and H. Zhang. Automatic browsing of large pictures on mobile devices. In *ACM Multimedia*, pages 148–155, Berkeley, CA, USA, Nov. 2003.

[12] P. Viola and M. J. Jones. Robust real-time face detection. *IJCV*, 57(2):137–154, 2004.

[13] L. Wolf, M. Guttmann, and D. Cohen-Or. Non-homgeneous content-driven video-retargeting. In *ICCV 2007*, Rio de Janeiro, Brazil, Oct. 2007.

[14] P. Yin, A. Criminisi, J. Winn, and I. Essa. Tree-based classifiers for bilayer video segmentation. In *CVPR 2007*, Minneapolis, MN, USA, June 2007.