CONTENT-AWARE COMPRESSION USING SALIENCY-DRIVEN IMAGE RETARGETING

Fabio Zünd[*†], *Yael Pritch*[*], *Alexander Sorkine-Hornung*[*], *Stefan Mangold*[*], *Thomas Gross*[†]

[*]Disney Research Zurich
[†]ETH Zurich

ABSTRACT

In this paper we propose a novel method to compress video content based on image retargeting. First, a saliency map is extracted from the video frames either automatically or according to user input. Next, nonlinear image scaling is performed which assigns a higher pixel count to salient image regions and fewer pixels to non-salient regions. The nonlinearly downscaled images can then be compressed using existing compression techniques and decoded and upscaled at the receiver. To this end we introduce a non-uniform antialiasing technique that significantly improves the image resampling quality. The overall process is complementary to existing compression methods and can be seamlessly incorporated into existing pipelines. We compare our method to JPEG 2000 and H.264/AVC-10 and show that, at the cost of visual quality in non-salient image regions, our method achieves a significant improvement of the visual quality of salient image regions in terms of Structural Similarity (SSIM) and Peak Signal-to-Noise-Ratio (PSNR) quality measures, in particular for scenarios with high compression ratios.

***Index Terms*—** video compression, image retargeting

1. INTRODUCTION

A large amount of live video content is consumed on mobile devices, typically via streaming over cellular networks. As the computational power of streaming servers and mobile devices is constantly increasing, the critical bottleneck remains the limited wireless channel capacity per device, in particular when each mobile device receives its own individual content independently (different camera views of events, individually selected replays, etc.). Wireless video streaming does not only rely on potentially high compression ratios but also demands high error-resilience of the transmitted data. Compression has been employed in all modern codecs as images contain significant statistical and visually subjective redundancy, and videos exhibit even more redundancy in between frames. This observation is the starting point for numerous approaches to reduce the size of an image while maintaining image quality [1]. Advanced video codecs such as H.264/AVC-10, which allow for high compression while maintaining high video quality, however, exhibit a raised sensitivity to data errors [2]. We propose content-aware compression using saliency-driven image retargeting (CCSIR) to integrate saliency maps into a compression pipeline. This method uses content-aware image retargeting to allocate more pixels to the important part of the image in a continuous, non-uniform way (see Fig. 1). The retargeting is followed by a multi-resolution approach in which different bands of the image are compressed with different ratios, using existing compression algorithms. An overview of current state of the art saliency compuation algorithms can be found in [11]. Note that the computation times are in the order of a few miliseconds. Hence, computing the saliency does not significantly increase the processing time of our suggested pipeline.

A basic form of CCSIR are existing region-of-interest (ROI) coding techniques which prioritize specific regions in an image. The JPEG 2000 standard [3] encodes certain regions at higher quality than the background. In the general ROI method, the wavelet transform coefficients in the ROI are scaled (shifted) so that their bits lie in higher bit-planes than the bits associated with the background. During the entropy coding the higher bit-planes are given higher priority and, therefore, the background is encoded in lower quality as the ROIs [4, 5, 6]. Further, [7] presents a method for ROI encoding in H.264/AVC similar to ROI encoding in JPEG 2000, and [8] presents an approach that blurs less salient regions using a foveation filter. With the high image frequencies removed, the non-salient regions can be compressed stronger.

These ROI methods are, however, strictly tied to a specific codec, whereas for CCSIR an arbitrary codec can be employed. Furthermore, our approach supports saliency maps representing arbitrary shapes rather than rectangular ROIs only, and the retargeting algorithm can be configured to generate smooth transitions from salient to non-salient regions, which is typically not the case for ROI based compression. Even though our experiments showed the best efficiency for a combination of CCSIR and JPEG 2000, our method is agnostic to the employed compression technique and hence complementary to existing approaches.

2. THE PROPOSED TECHNIQUE

Fig. 1 depicts an overview of the compression pipeline. First, the input image is downscaled (retargeted) to a smaller reso-

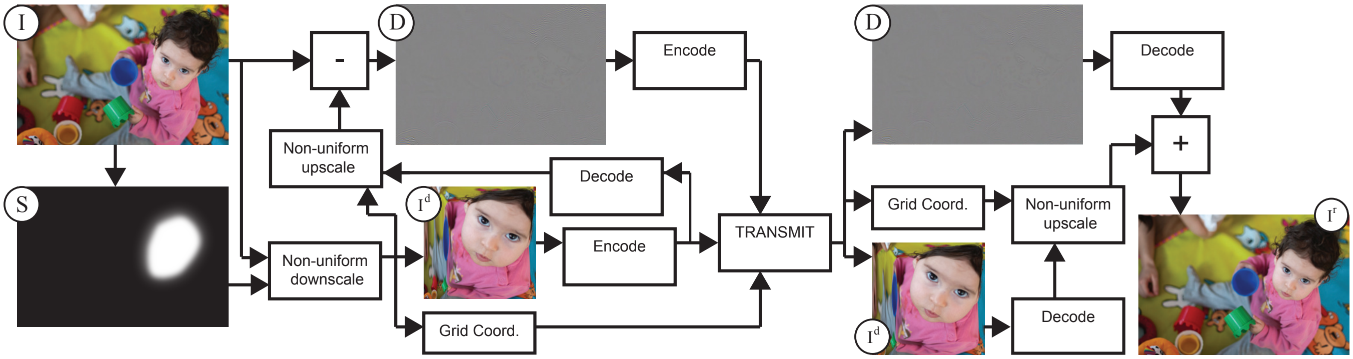


Fig. 1: Pipeline architecture: The input image I is downscaled to I^d according to the saliency image S. From I^d and I, a difference image D is created. Images are encoded to J2K and streamed. To decode, I^d is upscaled and D is added.

lution in a non-uniform way. Hence, more pixels are assigned to more salient areas of the image. While in principle any retargeting method can be employed, the recent axis-aligned retargeting algorithm of [9] is computationally particularly efficient and its warping technique is guaranteed an overlap-free bijective mapping. The scaling is based on saliency in a non-uniform way: Most of the reduction in resolution occurs in non-salient regions.

The downscaled image is then encoded with an arbitrary image encoder (JPEG 2000 in our example). To compensate for information loss that occurs during downscaling and encoding, a difference image is calculated, i.e., we compute a laplacian image pyramid [10] with a single level. The difference image contains the differences between the original input image and the downscaled and encoded image after it is upscaled back to its original size. It is then encoded as well. The total file size of the encoded image comprises the file size of the downscaled image, the difference image, and the set of grid coordinates, which is required to upscale the downscaled image back to its original shape.

From the input image I, we create a saliency map S (e.g., using [11]). Alternatively, in an interactive encoding system, the saliency map could be created by a user. When encoding the input image I, we create a set of three components $\mathrm{ENC(I)_S} = \{\mathrm{I^d, D, C}\}$, where $I^d = \mathrm{downscale(I)}$ represents the downscaled image, D is a difference image, and C is a set of grid coordinates. C is calculated solely from S by the retargeting algorithm. All three components are transmitted to the receiver. On the receiver, we decode the components into the reconstructed image I^r with $\mathrm{DEC(\{I^d, D, C\}) = I^r}$. If D is losslessy compressed, then $I = I^r$ holds, that is, the original image is perfectly reconstructed. By adjusting the compression level of I^d and D, we can control the quality of the compressions.

2.1. Encoding

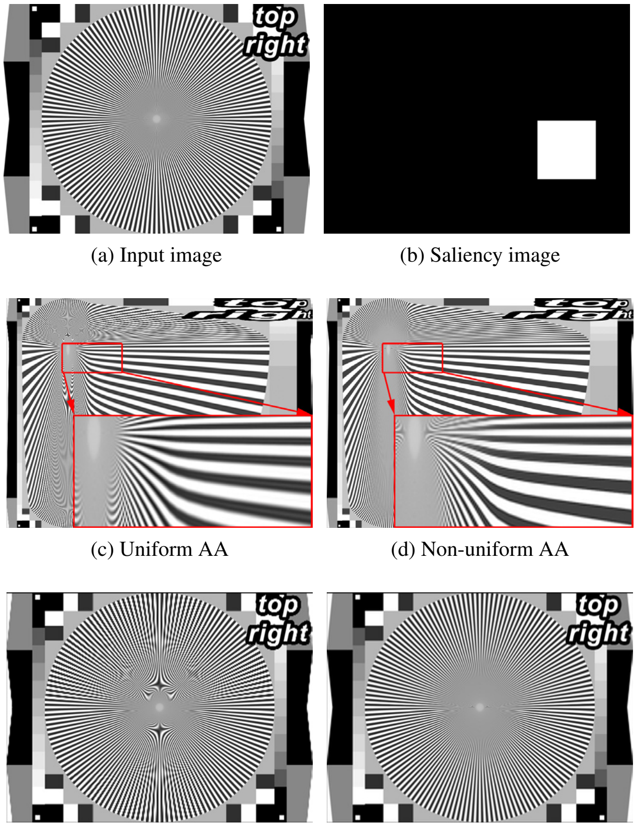
I^d is calculated by applying an image retargeting algorithm to the original image I, using S. Following [9], we overlay an uniform grid over the input image and we compute an axis-aligned deformed grid, which is calculation from the desired target scaling factor s and the saliency map S. A bicubic interpolation on the image is performed according to the deformed grid to scale the image down to the new resolution. The deformed grid coordinates C are saved along with the downscaled image I^d. I^d is then encoded using JPEG 2000. To create the difference image D, we decode I^d, upscale it again and calculate D, comprising all missing image content that was lost during the downscaling process as well as during the JPEG 2000 compression, i.e. it contains the JPEG 2000 compression artifacts. HFCR (high frequency compression ratio) denotes the JPEG 2000 compression ratio for D and LFCR (low frequency compression ratio) denotes the JPEG 2000 compression ratio for I^d.

2.2. Decoding

To restore the original image, we first decode the JPEG 2000 encoded images I^d and D and perform a bicubic interpolation on I^d, according to C, to upscale. Finally, we add D: $I^r = \mathrm{dec(D) + upscale_C(dec(I^d))}$. Note that even if we choose to highly compress I^d, we can, with a losslessy compression of D, perfectly reconstruct I.

2.3. Non-uniform Anti-Aliasing

Sampling theory dictates that, when subsampling a signal, the Shannon-Nyquist sampling theorem must be satisfied to prevent aliasing. We can prevent violation of the sampling theorem by applying an anti-aliasing filter, i.e., by attenuating the high frequency components. In CCSIR, as the image is subsampled in a non-uniform way, we need to apply a non-uniform low-pass filter. We employ the 6-tap cubic spline interpolation filter as described in [12], that approximates the Lanczos-3 kernel. Each axis is filtered independently. For every grid column, we calculate the scaling factor s_i^x, where $i \in [0, N]$, and s_j^y, where $j \in [0, M]$, respectively, based on the deformed grid with N columns and M rows. We di-



(a) Input image

(b) Saliency image

(c) Uniform AA

(d) Non-uniform AA

(e) Uniform AA

(f) Non-uniform AA

Fig. 2: Different anti-aliasing (AA) methods. (c): downscaled input image using uniform AA. (d): downscaled input image using non-uniform AA. (e): upscaled image (c). (f): upscaled image (d). Notice the artifacts in (c) and (e) that are avoided in (d) and (f).

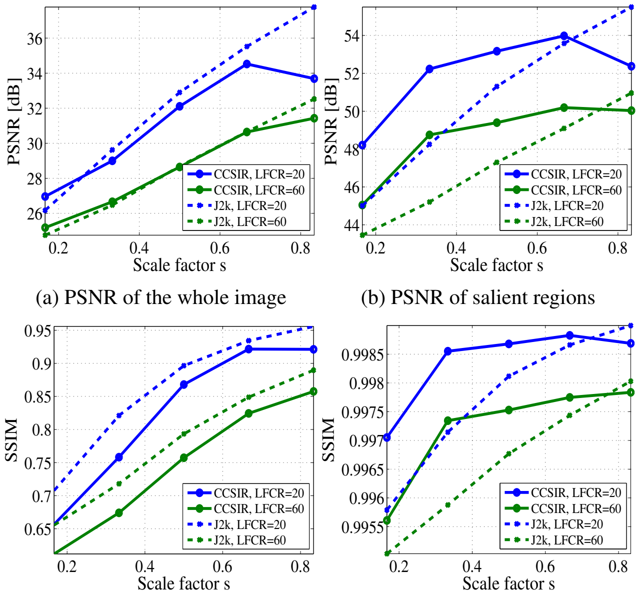
vide the scaling factors into segments, which correspond to a certain mean scaling factor value. A segment is marked if the absolute delta scaling factor exceeds a threshold, i.e., if $|\Delta s_i^x| > e$, holds, where $\Delta s_i^x = s_{i+1}^x - s_i^x$.

In our implementation, a threshold value of $e = 0.05$ is used. For every segment, the mean scaling factor is calculated and the 6-tap smoothing filter is applied on the corresponding part of the image. All parts are then linearly blended together. Fig. 2 compares our approach with uniform anti-aliasing. For this illustration we manually created the saliency image in Fig. 2b. The uniformly anti-aliased images Fig. 2c and Fig. 2e exhibit aliasing artifacts towards the middle of the star pattern and too much smoothing in the salient regions. The non-uniformly anti-aliased images Fig. 2d and Fig. 2f show significantly reduced aliasing.

3. RESULTS

A prototype Matlab implementation that performs downscaling and upscaling as described in Section 2 was created to evaluate CCSIR. The implementation relies on [9] for calculating the deformed grid. The evaluation images are retargeted with a high non-uniformity to differentiate our method from the other methods. In practice, a smoother transition from salient to non-salient regions could be targeted and then a moderate non-uniformity would suffice. To compare CCSIR we selected a compression ratio for the respective reference method so that the resulting file size equals our encoded image file size as close as possible. Given the same file size, we calculate two image quality metrics, Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity (SSIM) index [13], for all pixels in the images and for the pixels in the salient regions only. Fig. 3 compares PSNR and SSIM values of CCSIR to JPEG 2000 compressed images for different scaling factors s for two given LFCRs and a constant HFCR $= 1200$. The PSNR and SSIM values are averaged for a collection of five video clips.

The compression ranges from 0.037 bpp to 0.857 bpp with an average of 0.313 bpp. As expected, for both PSNR and SSIM, CCSIR performs slightly worse than JPEG 2000 on the overall image (Fig. 3a and Fig. 3c). However, it performs better in the salient areas of the images for moderate scaling factors. For large scaling factors $s > 0.75$, I^d is relatively large and the high total number of bits in I^d and D allow JPEG 2000 to compress with a relatively low compression ratio, and thus achieve higher quality.



(a) PSNR of the whole image

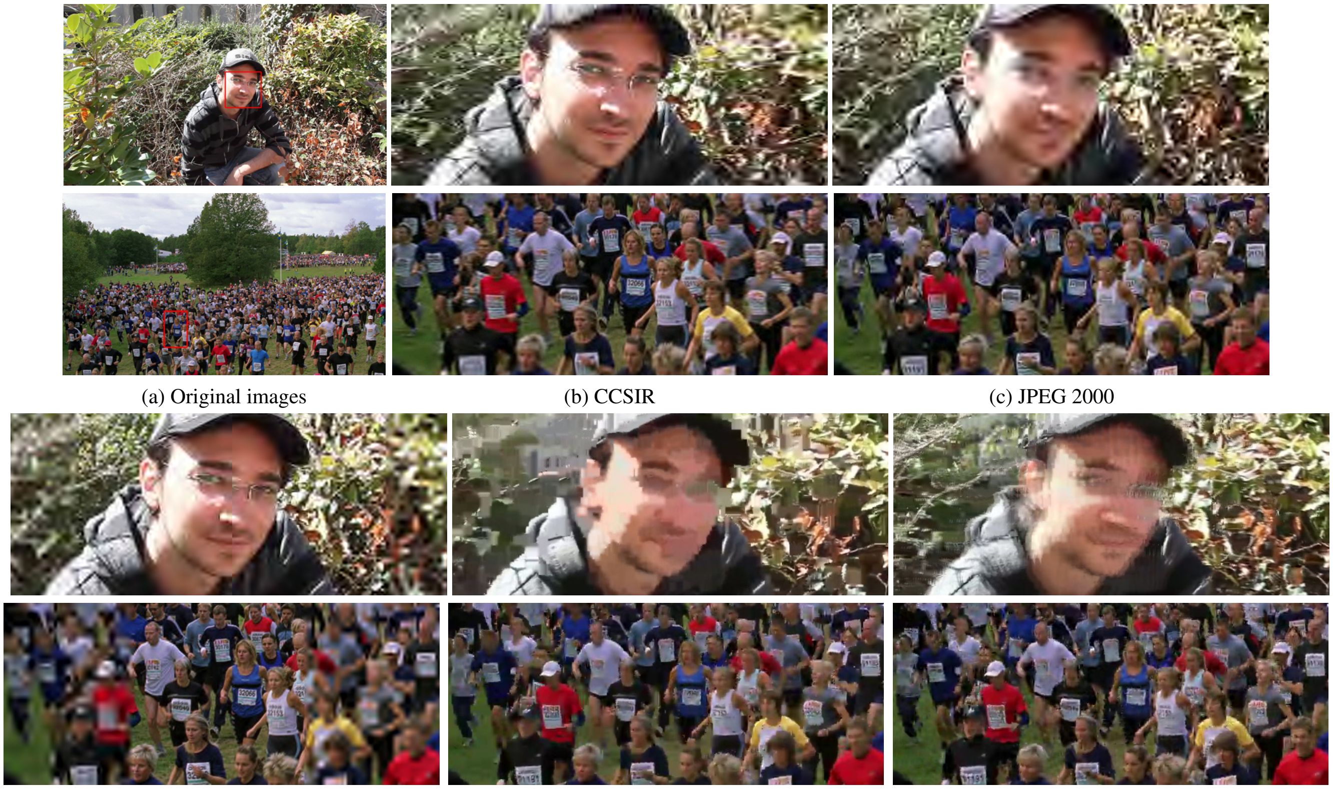
(b) PSNR of salient regions

(c) SSIM of the whole image

(d) SSIM of salient regions

Fig. 3: PSNR and SSIM overall and in salient regions only, for different scaling factors.

Fig. 4 provides a visual comparison to other known methods. One representative frame from each of two 100 frames video clips (*man* and *marathon*) is shown. The salient region is marked in red in the original images. For the saliency



(a) Original images

(b) CCSIR

(c) JPEG 2000

(d) JPEG 2000 ROI

(e) H.264 Baseline

(f) H.264 High

Fig. 4: Comparison to other methods. The red squares in (a) indicate the salient areas. All videos have equal file sizes. Each frame is compressed at approx. 0.07 bpp.

in the *man* images we automatically detect faces using [14]. The video encoding parameters are: LFCR $= 60$, HFCR $=$ 1800, $s = 1/3$ for the *man* video and LFCR $= 20$, HFCR $=$ 3000, $s = 1/3$ for the *marathon* video. The frames were compressed to approx. 15 KB (*man*) and to approx. 37 KB (*marathon*). The H.264 videos were encoded once using the High profile and once using the *Baseline* profile. The target bit rate for the H.264 videos was empirically determined to achieve a total file size that is equal to the total file size of the video encoded by CCSIR and the JPEG 2000 encoded video. Both H.264 videos were encoded using ffmpeg/x264 [15]. The JPEG 2000 ROI enabled video was encoded in Photoshop CS 5. The important regions are well recognizable in the CCSIR compressed frames. In the JPEG 2000 ROI frames, the regions are more precisely preserved. However, the transition from salient to non-salient regions is clearly visible, which is undesirable. One main advantage of CCSIR over JPEG 2000 ROI coding is that we can encode the image with an arbitrary smooth transition from salient to non-salient regions. We conducted a preliminary user study with 20 participants to assess the subjective quality of the five different versions of the *man* video. 75% of the participants rated (b) as the visually most appealing video, the next runner-up, (c) attracted 15% of the votes, followed by (d), (e), and (f). This result, although preliminary and subject to further validation, is encouraging.

The prototype implementation ran on an Intel i5 3.3 GHz Linux PC with 4 GB of RAM. We believe an implementation of CCSIR that benefits from current tuned software implementations or hardware accelerations for bicubic scaling, anti-aliasing, and JPEG 2000 encoding [16], could encode 24 fps 1080p video in real-time.

4. CONCLUSION

We presented an approach to content-aware saliency-driven video compression based on image retargeting. The approach exploits non-uniform anti-aliasing, which prevents aliasing in the highly scaled regions while avoiding over-smoothing in other regions. One attractive feature is that this approach can be easily incorporated into any existing compression pipeline.

Our method has most noticeable benefits if the source video contains a large amount of changes (e.g., due to object or camera motion) that result in many high frequency details. As the increasing interest for video content or competing video streams face the rim of capacity limits of wireless channels, content-aware compression provides a path to maintain quality in the critical regions while reducing storage and bandwidth demands.

5. REFERENCES

[1] T. Sikora, “MPEG digital video-coding standards,” *Signal Processing Magazine, IEEE*, vol. 14, no. 5, pp. 82–100, Sept. 1997.

[2] A. Kostuch, K. Gierssowski, and J. Wozniak, “Performance Analysis of Multicast Video Streaming in IEEE 802.11 b/g/n Testbed Environment,” in *Wireless and Mobile Networking*, Jozef Wozniak, Jerzy Konorski, Ryszard Katulski, and Andrzej Pach, Eds., vol. 308 of *IFIP Advances in Information and Communication Technology*, pp. 92–105. Springer Boston, 2009.

[3] C. Christopoulos, A. Skodras, and T. Ebrahimi, “The JPEG2000 still image coding system: an overview,” *Consumer Electronics, IEEE Transactions on*, vol. 46, no. 4, pp. 1103–1127, Nov. 2000.

[4] L. Liu and G. Fan, “A new JPEG2000 region-of-interest image coding method: partial significant bitplanes shift,” *Signal Processing Letters, IEEE*, vol. 10, no. 2, pp. 35–38, 2003.

[5] E. Atsumi and N. Farvardin, “Lossy/lossless region-of-interest image coding based on set partitioning in hierarchical trees,” in *Image Processing, 1998. ICIP 98. Proceedings. 1998 International Conference on*, Oct. 1998, vol. 1, pp. 87 –91 vol.1.

[6] Z. Wang and A. C. Bovik, “Bitplane-by-bitplane shift (BbBShift) - A suggestion for JPEG2000 region of interest image coding,” *Signal Processing Letters, IEEE*, vol. 9, no. 5, pp. 160–162, May 2002.

[7] Y. Liu, Z. G. Li, and Y. C. Soh, “Region-of-Interest Based Resource Allocation for Conversational Video Communication of H.264/AVC,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 1, pp. 134–139, Jan. 2008.

[8] L. Itti, “Automatic Foveation for Video Compression Using a Neurobiological Model of Visual Attention,” *IEEE Transactions on Image Processing*, vol. 13, no. 10, pp. 1304–1318, Oct. 2004.

[9] D. Panozzo, O. Weber, and O. Sorkine, “Robust image retargeting via axis-aligned deformation,” *Computer Graphics Forum (proceedings of EUROGRAPHICS)*, vol. 31, no. 2, pp. 229–236, 2012.

[10] P. Burt and E. Adelson, “The laplacian pyramid as a compact image code,” *Communications, IEEE Transactions on*, vol. 31, no. 4, pp. 532 – 540, apr 1983.

[11] F. Perazzi, P. Krähenbühl, Y. Pritch, and A. Hornung, “Saliency filters: Contrast based filtering for salient region detection,” in *CVPR*, 2012, pp. 733–740.

[12] Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG (ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6), “Upsampling Filter Design with Cubic Splines,” Apr. 2006.

[13] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: from error visibility to structural similarity,” *Image Processing, IEEE Transactions on*, vol. 13, no. 4, pp. 600–612, Apr. 2004.

[14] L. Wolf, T. Hassner, and Y. Taigman, “Effective Unconstrained Face Recognition by Combining Multiple Descriptors and Learned Background Statistics,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 33, no. 10, pp. 1978–1990, 2011.

[15] “x264,” www.videolan.org/developers/x264.html, viewed 2013-02-05.

[16] “Kakadu JPEG 2000 SDK,” www.kakadusoftware.com, viewed 2013-02-05.