An Approach to Video Compression Using Saliency Based Foveation

Adam Polakovič[1], Radoslav Vargic[1], Gregor Rozinaj[1], Gabriel-Miro Muntean[2]
[1]Slovak University of Technology in Bratislava, Ilkovičova 3, 812 19, Bratislava, Slovakia
[2]School of Electronic Engineering, Dublin City University, Glasnevin, Dublin 9,Ireland
adam.polakovic@stuba.sk

Abstract— **This research aims to increase quality of experience for video consumers by introducing an approach with saliency-based video foveation coupled with compression algorithm. This approach uses eye tracking information to build the saliency map and allows multiple usage scenarios. The approach can be used in single or multiple viewer environments. We evaluate the approach and provide results based on subjective measurements. The results confirm that the proposed approach for video compression is competitive and can deliver better quality of experience than standard video compression algorithms.**

Keywords—**Saliency; Eye tracking; Compression; Foveation**

I. Introduction

For variety of applications it is important to know which regions in the observed scene attend human visual system and are more salient for an observer. This information could be used for image and video compression or transmission as well. Saliency-aware video compression can encode parts of observed scene with higher bitrate than regions that don't attract human visual attention. This is thanks to the uneven distribution of photoreceptors in human retina. The corresponding decrease of resolution (or detail) from eye fixation point is called image foveation [1]. In this contribution we provide an approach for video compression that uses eye tracking to obtain gaze positions and in turn the saliency information. The saliency information is used in video encoding, where the foveation principle is implemented. The paper is organized as follows: In the next sections we introduce saliency and foveation in video compression in more detail. Next, we present the details of proposed approach, the approach is evaluated in part III and corresponding conclusion is provided in part IV.

A. Saliency

Salient regions are parts of observed scenes that attract human gaze. There are two main approaches used for salient regions detection: Eye tracking and visual attention modelling. Eye tracking is usually done with cameras used to record eye movements [2]. The goal in visual attention modeling is to detect locations attracting gaze and predicting behavior of human visual attention [3]. In case of video, location is determined not only by visual features of the object but also by objects movement. It turns out [4] that for video the quality of current modelling approaches is worse than even single observer with eye tracking. There were also proposed models [5] combining both eye tracking and visual attention models – a semiautomatic saliency model to take advantages from both approaches. The saliency information is usually expressed in the form of Saliency map, which is a grayscale image representing visual saliency of corresponding visual scene, having the same resolution as the original image. The values (usually 8-bit) represent how important (0=least important, 255=most important) is the certain pixel for an observer, i.e. how it attracts observers gaze. Saliency map can be constructed for each frame of the video and used in encoding process. The saliency information is also used in form of saliency map video [5]. The saliency information can be build offline or real-time.

B. Foveation and video encoding

The main goal of saliency-based encoding is allocating bits in favor of salient regions. There are two main approaches to implementing this idea:

- Modifying internal encoder data – e.g. setting different quantization parameter (QP) for macroblocks in salient regions from non-salient regions [6]. There exists e.g. implementation for x264 [5] which takes saliency map video as additional parameter, producing saliency-aware video with modified QP for macroblocks with different saliency significance. In comparison with regular x264 their model with their encoder proved to have 23% bitrate savings whilst having the same subjective and objective quality.
- Preprocessing before encoding –the aim is to reduce the amount of detail in non-salient regions before encoding, e.g. using non-uniform blurring [1]. The idea behind is that by removing high-frequency components from video frames in regions that are not attracting human visual system the video encoder with fixed bitrate will allocate more bits in favor of salient regions.

Both approaches introduce foveation, because in the non-salient regions the quality which corresponds to the level of detail or quality is reduced. However, the second approach is more straightforward related to foveation and allows easier split-up and separate analysis of both parts of the process.

Specific requirements arise, when we want to do the whole process in real-time, i.e. saliency information is build in real-time and announced to encoder. This use case has strict conditions to the latency of the whole loop which includes: eye-tracker delay, two times network delay, encoder delay and decoder delay. Moreover, it puts the restrictions to the encoding process (e.g. very limited number of B frames, buffer sizes, …). In this case it is important to study the time limits for foveation. In the context of foveated rendering the limits were studied for example in [7].

II. Proposed method

Our approach uses more straightforward way to reduce the bitrate in non-salient regions – preprocessing of the video frames before encoding. The process can be done in real-time as well. One typical use case (use case A) is that we want to create and store the saliency optimized video and have eye tracking data from multiple users watching the original video. Optimization can e.g. reduce the output video size while maintaining the same quality of experience (QoE). Another typical use case (use case B) is to perform the single user video delivery in real time. Display device has built-in the eye-tracker and in real-time give information to the encoder about user gaze position. The delivered video stream contains foveated video frames. For this use case it is important to study the time limits for foveation separately and build the environment where each part can be analyzed separately. Application of this use case are e.g. home (LAN) based multimedia delivery using VR or 5G networks multimedia delivery [8].

The proposed approach uses observed video sequence and corresponding eye tracking data as an input. As output it produces the saliency enabled video. Corresponding saliency map for certain video frame is computed from input gaze data. Then the frame is foveated, i.e. pre-processed by non-uniform blurring with strength corresponding to the saliency value of the certain pixel. The processed frames are then passed to the video encoder (x264). The whole process can be done in real-time as well, however simplified version is used in this paper, where all foveated video frames are gathered together and final saliency enabled video is produced afterwards.

A. *Generating saliency maps*

Original videos were taken and unfolded frame by frame. For each frame corresponding coordinates for each eye tracker added two-dimensional Gaussian function with the center in eye tracker coordinates, resulting values of pixels in saliency map were:

$$p(x,y) = \sum_{i=1}^{N} e^{-\frac{1}{2\sigma_h^2}\left(\frac{\left(x-x_{e_i}\right)^2}{\left(\frac{W}{H}\right)^2}+\left(y-y_{e_i}\right)^2\right)} \tag{1}$$

Where *p(x, y)* is the value of pixel on relative coordinates *x*, *y* with values from 0 to 1 (0 being the leftmost(x)/bottommost(y) pixel and 1 the rightmost(x)/topmost(y) pixel of the video frame), *i* is variable going from 1 to *N*, where *N* is the number of eye tracker coordinates, *e* is Euler's number, σ_h is standard deviation of relative height coordinates (*y*), W is the width of the video in pixels, H is the height of the video in pixels and x_{e_i} and y_{e_i} are relative coordinates of *i-th* eye tracker position. The saliency map now contains *N* overlapping two-dimensional Gaussian functions, generating values from *0* to *N*.

B. *Blurring non-salient regions*

Values from generated saliency map are taken and transformed to values from 0 to 1. The transformation needs to generate number 1 in highly salient regions (*h*) and number $\frac{1}{9}$ in lower salient regions (*l*). This number will be the middle pixel of Gaussian filter kernel and number 1 translates to zero standard deviation and number $\frac{1}{9}$ translates to infinite standard deviation of the Gaussian filter kernel. Suitable transformation function candidate is logarithm function with parameters *a*, *b*:

$$a + b \cdot ln\big(p(x,y)\big) \tag{2}$$

This produces two equations fulfilling conditions:

$$a + b \cdot ln(h) = 1 \tag{3}$$

$$a + b \cdot ln(l) = \tfrac{1}{9} \tag{4}$$

For smooth transition between salient and non-salient regions we have used values $h = \frac{N}{4}$ and $l = \frac{N}{8}$ with solution:

$$\frac{10}{9} + \frac{1}{18\cdot\log(2)} \cdot ln\left(\frac{p(x,y)}{N}\right) = p_n \tag{5}$$

Where p_n is the transformed pixel value, this number is calculated for each pixel, and is the center of 3x3 Gaussian filter kernel depicted in Fig. 1 that is applied on the pixel.

$\frac{p_n}{a^2}$	$\frac{p_n}{a}$	$\frac{p_n}{a^2}$
$\frac{p_n}{a}$	p_n	$\frac{p_n}{a}$
$\frac{p_n}{a^2}$	$\frac{p_n}{a}$	$\frac{p_n}{a^2}$

Figure 1. Gaussian filter kernel applied on each pixel.

Where *a* is (assuming the sum of all elements in matrix is 1):

$$a = 2 \cdot \frac{p_n + \sqrt{p_n^2 + (1-p_n)\cdot p_n}}{1-p_n} \tag{6}$$

This kernel is applied on the pixel, blurring has 20 iterations on the frame, meaning that each pixel will have this kernel applied 20 times. Blurred videos created by this process are then encoded with x264 encoder with constant bitrate. This whole workflow is demonstrated on examples and depicted in Fig. 2.

C. *Implementation*

Two implementations are created. One in Matlab to allow for more effective and precise evaluation and one in Unity to allow for easy real-time implementation in the VR environment. In Unity the operations with pixels are implemented using graphics card shaders for efficiency. The real-time implementation is ready to start the real-time experiments e.g. to simulate the effect of time delay in the foveation process in the use case B mentioned at the beginning of section II.

III. Evaluation of the proposed method

A. *Eye tracking and video data*

Eye tracking data and video data were provided by [5]. In our experiments we have used two 18 second video sequences.

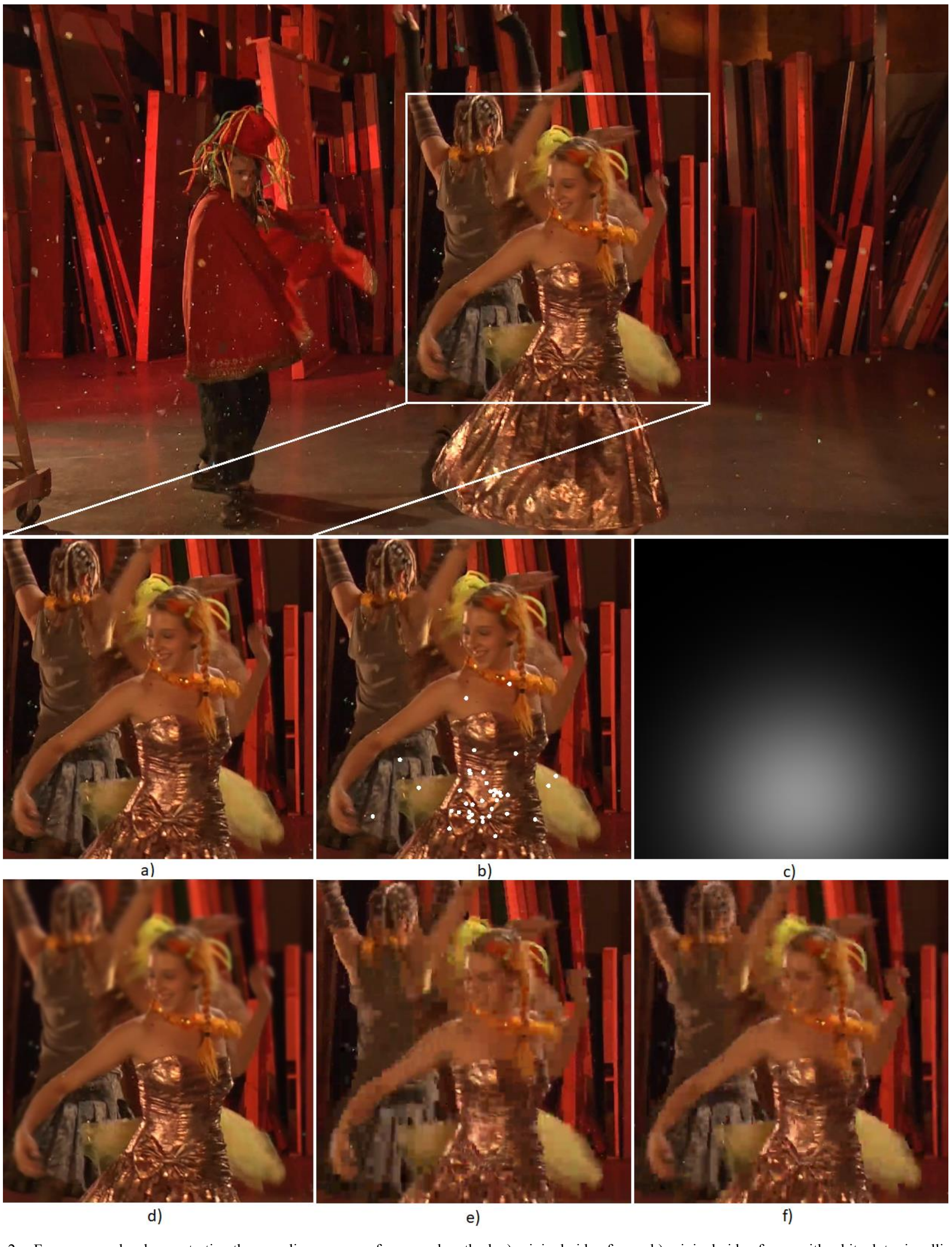

Figure 2. Frame examples demonstrating the encoding process of proposed method: a) original video frame, b) original video frame with white dots signalling eye tracker positons, c) saliency map created from eye tracker coordinates, d) blurring using saliency map, e) original video encoded with x264 codec with 1.5 Mbps, f) blurred video (σ_h = 0.15) encoded with x264 codec with 1.5 Mbps.

One video from a movie (Avatar, Directed by James Cameron. 2009) and one from LIVE Video database (Dance) [9]. Eye tracking data were averaged for each frame, so that each frame will have an array of eye tracking coordinates (in this case the size of array was 58). Two video sequences were blurred with method proposed in the section II using standard deviations $\sigma_h = 0.15$, $\sigma_h = 0.30$ in saliency model and without any blurring and then encoded with 1.5 Mbps in x264 codec. All together this makes up 6 videos (2 videos with 3 representations each).

B. Quality of experience experiments

44 participants aged between 21 and 61 participated in the experiment, each participant saw each video once and picked the best and worst video in terms of quality of experience. The results are in Table 1 and are displayed in the Fig. 3 and Fig. 4.

TABLE I. QUALITY OF EXPERIENCE (BEST METHODS MARKED GRAY)

	Avatar			Dance		
	Best	***Average***	***Worst***	***Best***	***Average***	***Worst***
Regular x264	4	6	34	15	14	14
$\sigma_h = 0.30$	11	25	8	18	20	5
$\sigma_h = 0.15$	29	13	2	10	9	24

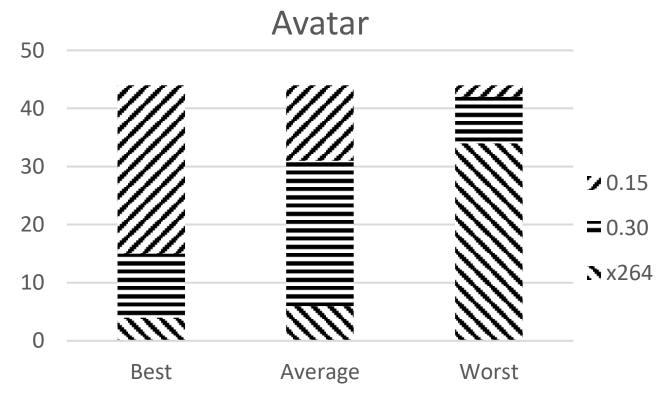

Figure 3. Quality of experience while watching Avatar video with different blurring strength ($\sigma_h = 0.15$, $\sigma_h = 0.30$) compared to regular x264 encoding.

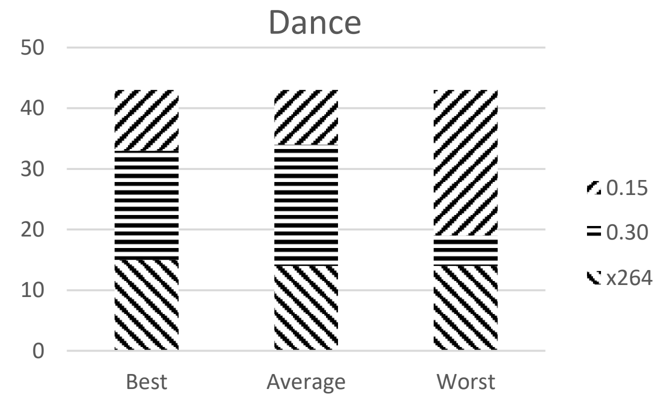

Figure 4. Quality of experience while watching Dance video with different blurring compared to regular x264 encoding.

In the case of Avatar video, the proposed method creates video with better quality of experience. Best video was with the smaller σ_h parameter and worst video was with regular x264 encoding. These results were evaluated with χ^2 test, in case of Avatar video, the χ^2 statistic was 74.73 and the corresponding p-value < 0.00001. In the case of Dance video from LIVE Video database, these results are less convincing with χ^2 statistic 19.12 and p-value 0.000746. The difference in results between these two videos are due to Avatar video having cuts once every two seconds, while the Dancing video has constant background, which needed to be encoded just once.

IV. CONCLUSIONS

In this paper we introduced a method for creating saliency maps from eye tracking data. We also introduced a method for pre-encoder blurring of non-salient regions in video to increase bitrate in salient regions. Our method was evaluated on video with fast cutting (3 seconds or less per shot) and slow cutting (whole video was made with one shot). We have proven that this method is an improvement in comparison to regular x264 encoder with the same bitrate and is suitable mainly for fast cutting videos.

ACKNOWLEDGMENT

The research described in the paper was financially supported by the H2020 project NEWTON, No. 688503 and VEGA project INOMET, No. 1/0800/16.

REFERENCES

[1] 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, October 2004.

[2] Y. W. Chen and K. Kubo, "A Robust Eye Detection and Tracking Technique Using Gabor Filters," Third International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP 2007), Kaohsiung, 2007, pp. 109-112.

[3] L. Itti, C. Koch and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 20, no. 11, pp. 1254-1259, November 1998.

[4] T. Judd, F. Durand and A. Torralba, "A benchmark of computational models of saliency to predict human fixations," Computer Science and Artificial Intelligence Lab, Massachusetts Institute of Technology, 2012.

[5] V. Lyudvichenko, M. Erofeev, Y. Gitman and D. Vatolin, "A semiautomatic saliency model and its application to video compression," 2017 13th *IEEE International Conference on Intelligent Computer Communication and Processing (ICCP)*, Cluj-Napoca, 2017, pp. 403-410.

[6] Z. Li, S. Qin and L. Itti, "Visual attention guided bit allocation in video compression," Image and Vision Computing, vol. 29, no. 1, 2011, pp. 1-14.

[7] R. Albert, A. Patney, D. Luebke, and J. Kim, "Latency Requirements for Foveated Rendering in Virtual Reality". ACM Trans. Appl. Percept. 14, 4, Article 25, September 2017, 13 pages.

[8] R. Vargic, M. Medvecký, J. Londák, P. Podhradský, "Advanced Interactive Multimedia Delivery in 5G Networks", 2018, pp. 421-430. 10.1007/978-3-319-75175-7_42.

[9] K. Seshadrinathan, R. Soundararajan, A. C. Bovik and L. K. Cormack, "Study of Subjective and Objective Quality Assessment of Video", *IEEE Transactions on Image Processing*, vol.19, no.6, June 2010, pp.1427-1441.