ENERGY-CONSTRAINED REAL-TIME H.264/AVC VIDEO CODING

Tiago A. da Fonseca and Ricardo L. de Queiroz

Universidade de Brasilia
Department of Electrical Engineering
Brasilia, DF, Brazil
tiago@image.unb.br, queiroz@ieee.org

ABSTRACT

Energy consumption has become a leading design constraint for computing devices in order to defray electric bills for individuals and businesses. Over the past years, digital video communication technologies have demanded higher computing power availability and, therefore, higher energy expenditure. In order to meet the challenge to provide software-based video encoding solutions, we adopted an open source software implementation of an H.264 video encoder, the x264 encoder, and optimized its prediction stage in the energy sense (E). Thus, besides looking for the coding options which lead to the best coded representation in terms of rate and distortion (RD), we constrain the process to fit within a certain energy budget. i.e., an RDE optimization. We considered energy as the time integration of the real demanded electric power for a given system. We present an RDE-optimized framework which allows for software-based real-time video compression, meeting the desired targets of electrical consumption, hence, controlling carbon emissions. We show results of energy-constrained compression wherein one can save as much as 35% of the energy with small impact on RD performance.

***Index Terms*—** Green computing, video codec, H.264/AVC, software implementation, tunable fidelity.

1. INTRODUCTION

Historically, processor manufacturers have responded to the demand for more processing power primarily with faster processor speeds. However, higher clock speeds imply in higher power consumption and heat. Image and video processing are driving forces behind this computational power pursuit. The *state-of-the-art* video compression standard, H.264/AVC [1]-[3], is a computation-hungry application. Nevertheless, energy usage and carbon emissions are a major concern today. Individuals, companies, and organizations move towards energy-efficient products as energy costs have grown to be a major factor [4]. Thus, saving energy has become a leading design constraint for computing devices through new energy-efficient architectures and algorithms.

Built as a hybrid DPCM encoder [5], the H.264 encoder searches for the best possible prediction of the encoding signal in order to provide the most compact representation [6]. However, that search is one of the most time consuming stages of a software-based video encoder [9]. Since encoders take most of time doing predictions, controlling the complexity of the prediction stage seems to be the natural way to control the overall complexity. In addition, when encoding video sequences in a low-latency communications applications, e.g. video conferencing, the time spent compressing the signal is an issue and real-time coding may be challenging. Traditionally, complexity can be considered as a measure of the effort to accomplish certain computation tasks and can be accounted either as the amount of memory, or the time, or the number of operations it takes to perform some computation [7]. We propose to evaluate energy demand instead of complexity, since energy is a fundamental resource that can be directly mapped to operational costs, and complexity estimation is not always a reliable indicator of energy consumption in multi-core platforms executing multi-threaded applications [8].

There are many works aimed at reducing AVC energy consumption by handling its complexity. Some explores prediction techniques for reducing computations with small RD penalties [10]-[12] but doesn't deal the energy expenditure directly as [13] does for its platform by handling specific features provided by the processor under test. A recent work yields a substantial H.264/AVC complexity reduction [14], however much of the complexity scaling will not be perceived if the framework is already implemented using faster algorithms, high-performance libraries and platform dependent resources [15],[16], which, by their turn, can be very energy demanding.

The present work suggests new strategies in the direction of saving energy in real-time computation. We present a fidelity-energy (ΦE) optimization strategy to constrain the energy demanded by an application in a real-time scenario; particularly for a software video encoder, the fidelity Φ can be evaluated in terms of the rate-distortion (RD) performance [5, 19]. Then, the optimized parameters are used to implement an RDE-optimized real-time encoding framework. We chose an open-source high-performance encoder, x264 [18], as the H.264/AVC software-implementation due to its excellent encoding speed and good rate distortion (RD) performance. The proposed approach suits, for example, mobile communication systems where energy efficiency is still a major bottleneck [17].

2. MEASURING ENERGY

Energy consumption is here measured in two ways. The computer is connected by itself (no monitor or other peripherals) to a wattmeter and from there to the local power supply. We can read the energy consumption from the wattmeter on another computer at every second, through a USB port. This is sufficient for steady state tests. However, in order to investigate the energy consumption behavior at very fast cycles (e.g. 30 Hz or 60 Hz video), which are comparable to the voltage cycles of the energy provided by our local power company (60 Hz), one has to resort to oscillography. For these tests we used an Elspec G4500 BlackBox and a California Instruments 5001ix sinusoidal power supply.

Time measurements can be disturbed by the the operating system (OS) scheduler in real-time systems. We used the Linux OS with kernel `2.6.32-3-amd64` and 250 Hz scheduler frequency. Considering the encoding speeds provided by our platform (a 6-core AMD® Phenom^{TM}-powered PC), the measurement of a short

This work was supported by CNPq under grant 470940/2010-7.

time intervals used to encode a video frame can be compromised. One way to overcome the scheduler induced variances is the grouping of frames in GOPs (Group of Pictures). The GOP grouping of frames also affects the demanded power waveforms. To illustrate this, we monitored our test platform, while compressing high-definition (720p 30-Hz) frames in real time, at different GOP sizes. As the processor is faster than necessary to guarantee real-time coding, the processor can "sleep" from the time it is done compressing a GOP until the next GOP is available for compression. The power waveform is registered by measuring the demanded power from the PSU's PC. The results from oscillography are presented in Fig. 1.

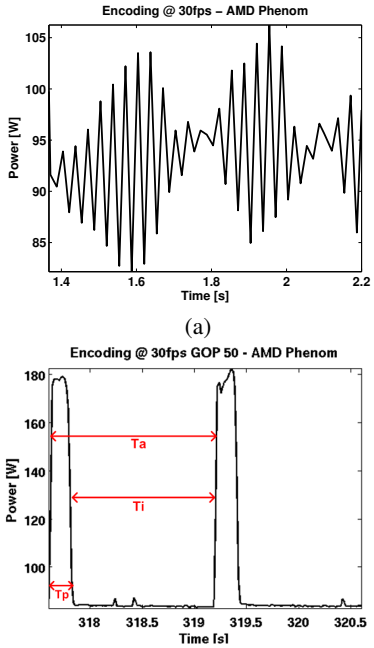


Fig. 1. Power waveforms for the encoding of 720p-video frames recorded and compressed at 30 Hz and grouped in different GOPs configurations: (a) 1-frame GOP; (b) 50-frame GOP. As the GOP size is increased, the waveform tends to a binary model. In (b), we highlight the time intervals of interest: frames are available in T_a intervals; demanded power is high while (T_p) the encoder is fully utilizing the processor; the processor returns to idle state reducing the energy consumption for T_i seconds waiting for a new frame.

The waveforms show distinctive GOP grouping signatures. The rapid processor commutation between the "idle" and the "processing" are discriminated in Fig. 1(a), where the power peaks represent the moment when the processor is fully busy encoding a frame, while the valleys represent a "sleep". The plot in Fig. 1(a) is a zoom of the process of compressing 24 frames. From the point of measuring (the PC's PSU), the "sleep" moments are not well determined, as the processor does not remain in the "idle" state (power demand of 80 W for our AMD® Phenom^{TM} processor) for a long period. This is the result of various factors: a filtering effect of the PSU related to the AC-DC conversion; the processor scaling due to DVFS (Dynamic Voltage and Frequency Scaling); and the ACPI activity over other PC components [8]. As the GOP size is increased (Figs. 1(b)), the waveforms approach the model of binary utilization swapping between a full-power state and an idle state. We arbitrate to use a 50-frame GOP to conform the waveform to a binary model and also avoid OS scheduling jitter in the time measurements required by our framework.

3. ENERGY-AWARE OPTIMIZATION

Typical optimization tasks deal with cost functions or success measures. Let a software encoder execute its job for which we can somehow measure its cost. For signal compression, the cost measure can be a measure of quality, like distortion (D) or the bit-rate (R) or a combination of both. The compression is assumed parametrized, i.e., one has the freedom to chose the values of N parameters $\{P_i\}_{i=1,\ldots,N}$. Let $\mathbf{P}$ be the vector with all P_i. The encoder runs on a given set of data Z that may be different at every instantiation. For every choice of $\mathbf{P}$ and Z, we can have a measure C of the encoder cost. In essence we can have a mapping

$$C = f(\mathbf{P}, Z).$$

Another attribute we can derive from each instantiation is the effort taken to execute the encoding task, which can be measured as demanded energy $E = g(\mathbf{P}, Z)$. It is expected that some parameters like number of iterations, data sizes, etc. would influence the demanded energy while some others would not. The central idea in this paper derives from the fact that the correlation of E and C is differently affected by different parameters. We will use this to find points that minimize the energy consumption. Specifically, we would like to operate in the lower convex hull (LCH, represented in Fig. 2 by green points), which is the set composed by instantiations that yield the lowest energy for a given cost. Departing from a simple explanation using a scalar cost, in video coding, the mapping is conveniently addressed by a multidimensional variable as $\mathbf{C} = [R, D]$. Hence, $\mathbf{C} = f(\mathbf{P}, Z)$.

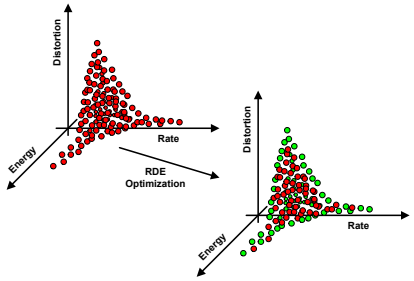


Fig. 2. Illustration on the set of RDE points that compose the Pareto front. The visible green points belong to the lower convex hull; some points are hidden due to the viewpoint.

$\mathbf{P}$ and Z are mapped to R, D and E, adding the energy dimension to the usual rate-distortion optimization problem.[1] We want to find the parameters that allow us to operate on the LCH in RDE space. In this manner, we can be assured that no configuration would yield lower energy consumption for a given cost value. Conversely, we can assure that, for a given energy consumption level, no other configuration would achieve better RD performance. Figure 2 illustrates the LCH in RDE space.

One approach is to use training data sets. Let $\{\mathbf{P}_k\}$ be the set of all parameter choices, ordered in some fashion. Let also $\mathbf{P}_k$ have elements P_{kn}. If we use a representative data set $\hat{Z}$, we can span $\{\mathbf{P}_k\}$, computing E, R and D for each choice and identifying the points that belong to the LCH of $E \times R \times D$. If the n-th point belongs to the LCH, we record $\mathbf{Q}_n = [E_n, R_n, D_n, \mathbf{P}_n]$, which contains the optimal points for the set $\hat{Z}$, but which are also assumed good enough for other data. The off-line training algorithm is:

1. Input a representative data set $\hat{Z}$ and create an empty list Q.

[1] We measure active power from which we can derive accumulated energy consumption.

2. For all k, compute $E_k = g(\mathbf{P}_k, \hat{Z})$ and $[R_k, D_k] = f(\mathbf{P}_k, \hat{Z})$. If point belongs to LCH, record $\mathbf{Q}_k = [E_k, R_k, D_k, \mathbf{P}_k]$ into Q.
3. Output a list Q of points in the LCH.

After finding the N_q points which belong to LCH, we sort Q in an ascending order of energy, i.e. $\{E_i\}$ in Q in non-decreasing. When running on-line, the parameter finding algorithm is as follows. Initially, consider a bit-rate R^r (channel constraint) and a desired energy target E^r. Then:

1. Input a list Q of points in the LCH, the energy target E^r and the rate target R^r. Create an empty list L.
2. Span Q, for $k = 1, \ldots, N_q$. If $|R_k - R^r| < \epsilon$ insert $\mathbf{Q}_k$ into L.
3. Count N_l, the number of itens in L. Note that the itens in L are still in ascending order of energy and all parameters are supposed to achieve similar bit-rate.
4. Span L, for $k = 1, \ldots, N_l$, until $E_k \leq E^r \leq E_{k+1}$, then stop.
5. Find $\mathbf{P}'$ as a proportional interpolation of $\mathbf{P}_k$ and $\mathbf{P}_{k+1}$ in L.
6. Output parameter vector $\mathbf{P}'$.

Parameter set $\mathbf{P}'$ is then used to compress data set Z. We used energy targets E^r constrained to a bitrate R^r, but it is trivial to replace it with a distortion target D^r. Of course, many parameters do not assume continuous values and some action has to be taken to properly assign them. For example, being that the case for the m-th parameter, one can use the value from P_{km} if $E^r - E_k < E_{k+1} - E^r$, otherwise use the value from $P_{k+1,m}$.

If a feedback control is allowed, one can monitor the system energy consumption and continuously adjust the parameters. If the energy consumption is not as predicted, it is because of discrepancies between Z and $\hat{Z}$, so that $\hat{Z}$ is not as representative as one would assume. Such a mismatch may also depend upon the non-linear mapping g. One solution is to start with a target E^r and to periodically measure the energy $E(n)$. We then adapt the parameters in order to control the energy expenditure (or cost). Assume that at any given instant n, $\mathbf{P}'$ is taken somewhere as an interpolation of $\mathbf{P}_j$ and $\mathbf{P}_{j+1}$. If $E(n) < E^r$ one should move $\mathbf{P}'$ towards $\mathbf{P}_{j+1}$ or even $\mathbf{P}_{j+2}$. Conversely, if $E(n) > E^r$ one should move in the opposite direction, i.e. towards $\mathbf{P}_j$ or even $\mathbf{P}_{j-1}$.

The control loop enjoy all the properties of trivial adaptive systems and there are many techniques to chose adaptation steps and to deal with convergence issues.

4. THE APPROACH

In order to scale the amount of energy used to encode a particular video sequence, we choose to modify the predictions stage, one of the most computational intensive steps when encoding digital video [9]. Our framework for energy-controlled real-time video coding consists in controlling the amount of energy spent while encoding a video sequence by adjusting the encoder in such a way that the RD penalties, compared to a full-featured case, remain as low as possible. Here, the energy (E) is measured by integrating on time the active power readings provided by a wattmeter attached to the power supply unit of a 6-core AMD® Phenom^{TM} desktop computer that executes the encoding tasks. We chose x264 as the H.264/AVC software-implementation due to its encoding speed and good RD-performance [18]. Using the methodology discussed on Section 3, our approach extends an RD-optimization [19] strategy, adding the E dimension (which stands for energy) to the regular 2-D problem of optimizing a particular codec to spend the smallest bitrate (R) necessary to represent a encoded video signal at a particular distortion (D).

Let $\mathbf{P}$ be the aggregation of the following computational-effort related parameters: the number of B-frames ($\#B$), the number of references frames (#Refs), the motion vectors precision (MVP) used in motion compensation, the chosen mode decision heuristic (MD), the quantization parameter (QP) and the number of encoding threads (#Thrds):

$$\mathbf{P} = \{\#\text{B}, \#\text{Refs}, \text{MVP}, \text{MD}, \text{QP}, \#\text{Thrds}\}\,.$$

The cost $\mathbf{C}$ is taken as the duple rate-distortion RD, where rate is taken as bit per seconds while the distortion is evaluated as the MSE (Mean Squared Error) between the original and the coded signal.

We opted to evaluate an empirical approximation to the energy function $E = g(\mathbf{P}, Z)$ through energy measurements. So, for each particular encoder setup, we compute the total bitrate (R), the MSE (D) and the energy spent to encode a training sequence (E). The RDE points are used to populate an initial search space from whose points that lie on its lower convex hull are derived by RDE optimization. After finding the setups that belong to the LCH, we build a lookup table from the performance numbers in order to provide optimal starting RDE points. Any intermediate demanded energy point not found in the table can be easily achieved by interpolation in such a way that the global demanded energy is very close to the energy "budget".

5. RESULTS

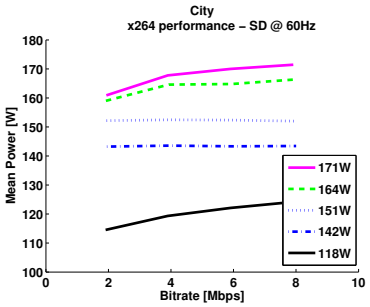
At every sequence that is compressed we obtain an RDE triplet. In order to display results in 2D, we can use the RD plots as in Fig. 3(b), one curve for each energy (power) level. It is important to note that not all points in a curve indicate the same power consumption. We simply labeled the curve by its average as shown Fig. 3(a), which indicates the actual power consumption as the controller tracks the demanded energy target for various bit-rates.

The curves in Fig. 3 are close to each other. In order to compare them, it is convenient to analyze averaged PSNR differences between two RD curves as described in [20]. For each sequence, each RD-curve is compared against the best RD-performance setup, which, in turn, has the highest averaged power expenditure. Power expenditure is also presented in relative numbers. The averaged results are illustrated in Fig. 4(a) for SD video sequences. The general behavior suggests that, as we reduce the available power (and energy) used to encode a video sequence, the performance penalties increase. In Fig. 4(b) the results are shown for 720p sequences.

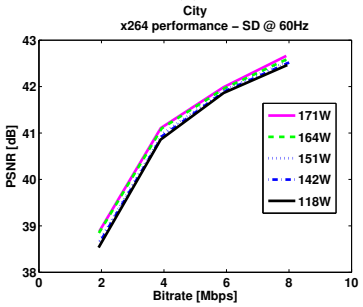
The main result is an energy-controlled framework which allows the user to choose the desired energy budget while real-time encoding HD and SD[2] video sequences. As expected, the RD-perfomance tends to be penalized as the encoding speed is raised. However, the curves are close to each other and the worst case is represented by high-motion high-frequency (50hZ) detailed sequences ("Shields" and "Mobcal"). For less demanding video sequences, like videoconferencing 30Hz ones ("Seq15" and "Seq21")[3], the framework penalizes the PSNR in less than 1.3dB on average while providing up to 35% of mean power and energy savings. The SD results, besides the increased baseline compression speed for real-time coding (60Hz), delivered lower PSNR drops (less than 0.6dB) for similar energy

[2] "Soccer" is present in the training and in the evaluation steps; however, the frames used to evaluate the encoder are from a different set from those used to build the training sequence.

[3] "Seq15" and "Seq21" are scenes where there is a couple of speakers on a table: the background is plain on "Seq15" and is detailed on "Seq21".

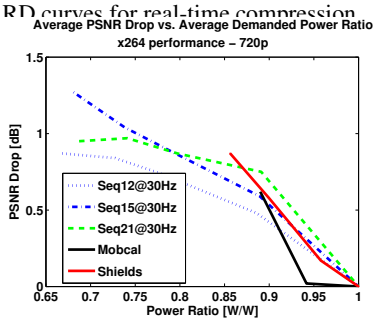


(a)

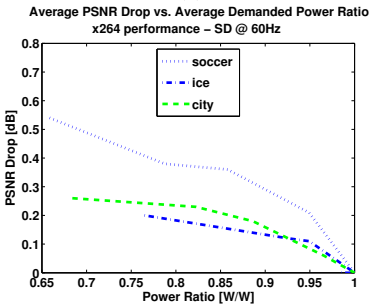


(b)

Fig. 3. Energy scaling for compressing SD sequence "City". A range of 10% of deviation is allowed for both bitrate and power. (a) Actual demanded power for various bit-rates and several target power demands. (b) RD curves for real-time compression



(a)



(b)

Fig. 4. PSNR drop vs. mean power ratio for (a) SD and (b) 720p video sequences. Video quality increases as we increase the power budget. A energy ratio of 1.0W/W represents the case of best RD-performance for real-time coding.

savings, even for very detailed video sequences. Better training sets may also lead to better results.

6. CONCLUSIONS

We proposed a fully-compliant energy-optimized framework for a H.264/AVC software implementation that allows for real-time coding. Rather than using all prediction tools provided by implementation, we can optimally choose a subset of them constrained by the amount of available energy. Our tests have shown that the RD performance is moderately affected by the framework, which does not make use of frame-skipping to comply to the requested energy budget while real-time encoding. Nevertheless, we provide significant demanded energy reduction.

In addition to the fact that out framework can be readily used to build PC-based video encoder appliances without the need of changing decoder implementation, our contribution can benefit from the availability of powerful computers for designing PC-based appliances. We plan to further work on making an encoder aware of environmental and communications conditions, capable of adjusting itself to meet channel, quality and energy constraints.

7. REFERENCES

[1] JVT of ISO/IEC MPEG and ITU-T VCEG, "Advanced video coding for generic audiovisual services," Tech. Rep. 14496-10:2005, March 2005.

[2] I. E. G. Richardson, *H.264 and MPEG-4 Video Compression*, John Wiley & Sons Ltd, 2003.

[3] R. L. de Queiroz, R. S. Ortis, A. Zaghetto, and T. A. Fonseca, "Fringe benefits of the H.264/AVC," in *Proc. International Telecommunication Symposium*, pp. 208–212, 2006.

[4] S. Albers, "Energy-efficient algorithms," *Communications of the ACM*, vol. 53, no. 5, pp. 86–96, 2010.

[5] K. Sayood, *Introduction to Data Compression*, Morgan Kauffmann Publishers, 2000.

[6] J. Ostermann, J. Bormans, P. List, D. Marpe, M. Narroschke, F. Pereira, T. Stockhammer, and T. Wedi, "Video coding with H.264/AVC: tools, performance, and complexity," *IEEE Circuits and Systems Magazine*, pp. 7–28, Jan-Mar 2004.

[7] Fortnow, L. and Homer, S., "A short history of computational complexity," *Bulletin of the EATCS*, vol. 80, pp. 95–133, 2003.

[8] Beloglazov, A. and Buyya, R. and Lee, Y.C. and Zomaya, "A taxonomy and survey of energy-efficient data centers and cloud computing systems," *Advances in Computers*, vol. 82, no. 2, pp. 47–111, Apr. 2011.

[9] Y.-Y. Huang, B.-Y. Hsieh, S.-Y. Chien, S.-Y. Ma, and L.-G. Chen, "Analysis and complexity reduction of multiple reference frames motion estimation in H.264/AVC," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 4, pp. 507–522, Apr. 2006.

[10] Z. Chen, P. Zhou, and Y. He, "Hybrid unsymmetrical-cross multi-hexagon-grid search strategy for integer pel motion estimation in H.264," *Proc. Picture Coding Symposium*, Apr. 2003.

[11] B.G. Kim, S.-K. Song, and C.-S. Cho, "Efficient inter-mode decision based on contextual prediction for the P-slice in H.264/AVC video coding," *Proc. IEEE International Conference on Image Processing*, pp. 1333–1336, September 2006.

[12] C. S. Kannangara, Y. Zhao, I. E. Richardson, and M. Bystrom, "Complexity control of H.264 based on a Bayesian framework," *Proc. Picture Coding Symposium*, Nov. 2007.

[13] Z. He, Y. Liang, L. Chen, I. Ahmad, and D. Wu, "Power-rate-distortion analysis for wireless video communication under energy constraints," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 15, no. 5, pp. 645–658, May 2005.

[14] M. Moecke and R. Seara, "Sorting rates in video encoding process for complexity reduction," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 20, no. 1, pp. 88–101, 2010.

[15] Kyu Park, Nitin Singhal, Man Hee Lee, , and Sungdae Cho, "Efficient design and implementation of visual computing algorithms on the GPU," *Proc. IEEE Intl. Conf. on Image Processing*, pp. 2321–2324, Nov. 2009.
[16] Intel, "Intel Integrated Performance Primitives," *http://software.intel.com/en-us/intel-ipp/*.
[17] O. Silven and K. Jyrkkä, "Observations on power-efficiency trends in mobile communication devices," *EURASIP Journal on Embedded Systems*, vol. 2007, no. 1, pp. 17–27, 2007.
[18] L. Merrit and R. Vanam, "Improved rate control and motion estimation for H.264 encoder," *Proc. IEEE International Conference on Image Processing*, vol. 5, 2007.
[19] Gary J. Sullivan and Thomas Wiegand, "Rate-distortion optimization for video compression," *IEEE Signal Processing Magazine*, vol. 15, no. 6, pp. 74–90, Nov. 1998.
[20] G. Bjontegaard, "Calculation of average PSNR differences between RD-curves," *VCEG-M33*, April 2001.