Energy-aware Complexity Adaptation for Mobile Video Calls

Haiyang Ma[†], Deepak Gangadharan[‡], Nalini Venkatasubramanian[§], Roger Zimmermann[‡]
[†] Graduate School for Integrative Sciences and Engineering,[‡] School of Computing
[†‡]National University of Singapore, Singapore
[§] Dept. of Computer Science, University of California, Irvine
[§]nalini@ics.uci.edu
[†‡]{haiyang,gdeepak,rogerz}@comp.nus.edu.sg

ABSTRACT

High energy consumption has become a challenge for multimedia applications on mobile platforms. We propose a cross layer framework that integrates complexity adaptation and energy conservation for mobile video calls. First we select the most utility-aware encoding and decoding parameters for videos of different motion levels through extensive offline profiling and analysis. Then we design a feedback algorithm to adaptively apply different coding parameters while monitoring the system performance online during a video call. To minimize energy consumption, we utilize Dynamic Voltage and Frequency Scaling (DVFS) for the CPU and buffered transmissions for the network. Our experimental results show an effective saving on energy consumption, with on average 52% savings on the CPU and 30% on the wireless network, while still maintaining high quality service.

Categories and Subject Descriptors

C.3 [**Special-purpose and application-based systems**]: Real-time and embedded systems; C.4 [**Performance of Systems**]: modeling techniques

General Terms

Algorithms, Design, Experimentation

Keywords

Energy-use optimization, adaptation, mobile video calls

1. INTRODUCTION

In recent years we have witnessed the explosive increase and widespread adoption of mobile devices such as smartphones and tablet PCs as entertainment hubs where multimedia applications have been emerging as a dominant usage pattern. For such applications sufficient processing power needs to be provided and a specific level of QoS (Quality of Service) should be sustained. As a result, they pose huge burdens for mobile devices with only constrained resources such as battery capacity, memory and processing capability.

Recently video calling has been introduced into mobile platforms, which we refer to as *Mobile Video Calls*, such as Apple's FaceTime. Compared with traditional multimedia applications, mobile video calls entail both an encoder and a decoder in simultaneous execution on energy-consuming hardwares such as CPU, graphical display, wireless network interface card (NIC) and integrated camera. Therefore mobile devices are more susceptible to fast draining of the battery and severe degradation in QoS. While past research has concentrated on workload reduction for either an encoder or a decoder, complexity adaptation of dual coding algorithms on mobile platforms has not been explored yet.

We propose an integrated framework based upon complexity adaptation and energy conservation for mobile video calls. We explore the complexity-quality design space for MPEG-4 [1] through detailed profiling and choose the most utility-aware coding parameters with regard to the video's context changes in a mobile scenario. We design a feedback-based adaptation mechanism where execution conditions of coding work, such as video quality, buffer size, coding delay, etc., are continuously fed back. In this way we manage to sustain QoS while considerably reducing the workload and energy consumption of mobile video calls.

2. RELATED WORK

Various methods to reduce the workloads of different functional components of video coding have been proposed. For the decoder, Peng [6] suggests properly pruning the DCT data within macroblocks to scale the computational complexity. Ji *et al.*[5] propose substitution of pixel interpolation modes in motion compensation. The complexity of the decoder could also be adapted in cooperation with the encoder, as put forth by Wang *et al.*[9] who adopt decoding-complexity aware encoding. For encoding, most research efforts have been devoted to motion estimation scaling. Ji *et al.*[4] propose to decrease the search range and motion vector resolution for complexity reduction. Shafique *et al.*[7] adopt a pixel decimation pattern and adaptive stopping criteria. Zhou *et al.*[11] propose a fast inter mode decision to curtail the set of possible coding modes.

Power management for resource-demanding multimedia applications on mobile platforms has long been an impor-

[*]Area Chair: Pal Halvorsen

tant research focus. Some researchers try to minimize energy consumption on hardware components, such as Cao *et al.*[2] who designed a DVFS algorithm for the CPU and, Tamai *et al.*[8] who proposed to shorten the working time of the network card using periodic bulk transfer. Some researchers seek the best tradeoff between QoS and energy consumption so that battery life can be extended without sacrificing much service quality. Typical methods include the one by Yuan *et al.*[10] who trade off multimedia quality against energy by leveraging a hierarchy of global and internal adaptations and, Hsu *et al.*[3] who utilize scalable video coding in video streaming service. However, these approaches either focus on one particular hardware component or coding functionality, and concurrent encoder and decoder optimization has not been considered. In contrast, our work provides a solution for a dual coding scenario with energy savings contributed by multiple components.

3. SYSTEM OVERVIEW

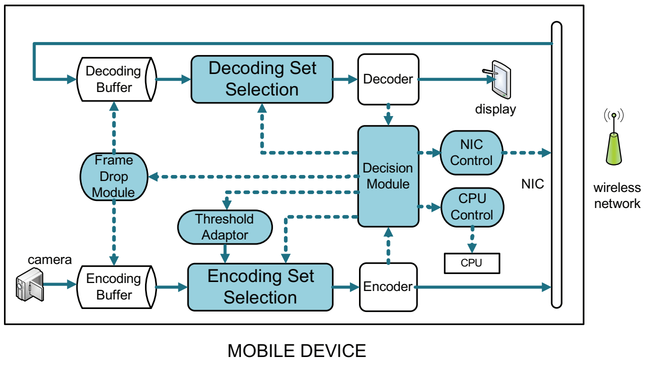


Figure 1: System framework. The white blocks illustrate the coding module and the solid arrows show its workflow. The blue (dark) blocks illustrate the proposed feedback module with its workflow indicated in dashed arrows.

The structure of our video call system on a mobile device is shown in Figure 1. The white blocks illustrate the coding module with its workflow indicated in solid arrows. Like other video conferencing systems, the coding module has an encoder and a decoder. The encoder continuously reads frames from the front-facing camera and encodes them into a compressed bitstream that is to be transmitted to the receiver through a wireless network. The decoder, on the other hand, reads the received bitstream from the network and decodes it into complete frames for display on the screen.

The blue (dark) blocks show the feedback module proposed in this paper and the dashed arrows indicate its workflow. The feedback module consists of a Frame Drop Module that decides when and how to drop frames, a Threshold Adaptor as well as an Encoding and a Decoding Set Selection module that decide which coding parameters should be selected for the current frames. The CPU and the NIC Control Module implement DVFS and buffered transmissions. The Decision Module controls all other modules' behaviors. Next we take a deeper look at the coding module for the workload reduction while the control mechanism and algorithm for the feedback module will be explained in Section 5.

4. VIDEO CODING COMPLEXITY SCALING

Most video codecs adopt a macroblock-oriented motion compensation approach and can be decomposed into several modules. The complexity of each module can be fine-tuned with different control parameters, generating videos of various qualities at different workloads. In order to explore the quality-complexity (energy) design space, we conducted extensive experiments and profiled the output and workloads of the main functional modules with various coding options. We utilize Xvid, an open source implementation of MPEG-4, as the video codec for analysis and experiments. We selected ten 150-frame video sequences in QCIF (176x144) format from a set of standard MPEG test videos[1]. We ran all profiling work on the SimpleScalar platform configured with the MIPS instruction set. The results are discussed in the following parts separately for the encoder and the decoder.

4.1 Encoder Adaptation

Motion Estimation (ME) is the most computationally intensive component of the encoding process, therefore we focus on the workload scalability in ME. During motion estimation, the encoder searches through the reference frames, trying to find a best matching motion vector (MV) for the current macroblock in terms of minimal pixel SAD (Sum of Absolute Difference). We define Motion Level $MotionL$ to be the average SAD between two consecutive raw frames, and show that video sequences with similar $MotionL$ show comparatively similar workloads under the same encoding parameters. For a frame, the workload of the ME module is controlled by three parameters: the search precision, the search range and the compensation mode. In Table 1 we list all the workload reduction options for encoding in MPEG-4 and there are 12 combinations in total, each of which is referred to as an *encoding set*. To select the most appropriate encoding set for different videos, we encode the ten test videos with all encoding options and record the resulting complexity and quality. To quantitatively evaluate the tradeoff between quality and complexity we define a utility function $Util$:

$$Util = \frac{\overline{\Delta COUNT}}{\overline{\Delta SAD}}.$$

$\overline{\Delta SAD}$, the average percentage by which the final matching SAD value increases compared to the original approach in Xvid, serves as the indicator of quality loss due to simplification in the ME process. Similarly, $\overline{\Delta COUNT}$ is the average reduction in the number of performed searches and SAD computations per macroblock. After processing all videos we observe that videos with similar motion levels tend to have similar performances for the same encoding options and the selected encoding sets fall into three clusters. As a result, we create three sets L1, L2 and L3 as listed in Table 2, which provide the best optimizations for video frames with high, medium and low $MotionL$ values.

4.2 Decoder Adaptation

Like the encoder, the decoder can also be decomposed into several components and the receiver may choose to simplify or ignore some components to decrease the workload. The primary mechanisms are Huffman Coefficient Discard and

[1]media.xiph.org/video/derf/

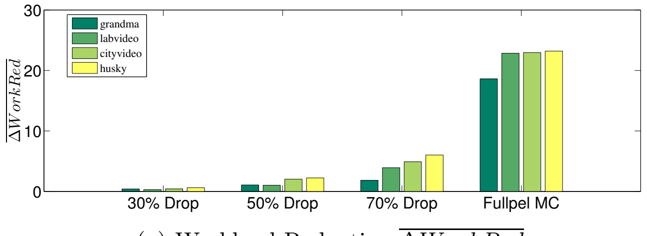
Item	Parameter
Search Range	original, half range, quarter range
Precision	fullpel, half and quarter pel
Coding Mode	all, DIRECT and INTER only

Table 1: Available encoding options.

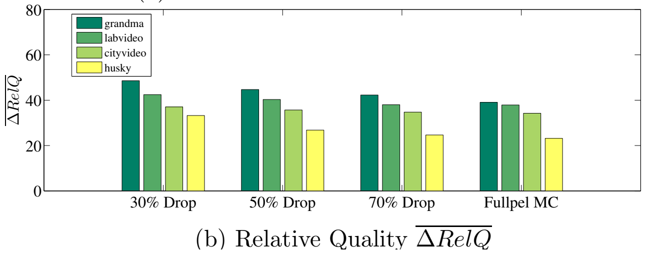
Level	Range	Precision	Mode	MotionL
L1	full	subpel	all	high
L2	half	fullpel	Direct,Inter only	medium
L3	quarter	fullpel	Direct,Inter only	low

Table 2: Selected encoding sets for adaptation.

Fullpel Compensation Replacement. To test the efficacy of these two approaches, we first decode the ten test videos with the standard decoding process (SDP) where no workload reduction measure is taken. We refer to this decoding set as D0. Then four decoding optimizations are tested, namely 30%, 50% and 70% Huffman codes drop and Fullpel MC replacement, respectively. The workloads are measured as average processor cycles per frame, and $\overline{\Delta WorkRed}$ denotes the percentage of the workload reduction compared to D0. For the decoder, we define the relative quality difference $\overline{\Delta RelQ}$ to be the average PSNR between the output frames produced by SDP and the complexity reduction measures mentioned above. An upper bound of 50 dB is set for output frames of identical content. We select four videos from our test set with different $MotionL$ (*grandma*: 2.57; *labvideo*: 9.91; *cityvideo*: 19.76; *husky*: 26.03). The decoding results of $\overline{\Delta WorkRed}$ and $\overline{\Delta RelQ}$ are depicted in Figure 2.



(a) Workload Reduction $\overline{\Delta WorkRed}$



(b) Relative Quality $\overline{\Delta RelQ}$

Figure 2: Workload Reduction $\overline{\Delta WorkRed}$ (a) and Relative Quality $\overline{\Delta RelQ}$ (b) for different decoding sets.

We can see from Figure 2 that when dropping Huffman codes the workload reduction rarely exceeds 5%, while Fullpel MC can reduce the workload by about 20% on average. Therefore, we will adopt Fullpel MC as the only workload reduction approach at the decoder side and refer to this decoding set as D1.

5. DECISION MODULE

Algorithm 1 illustrates the workflow of the decision module. Once a frame's coding work has been completed, its corresponding statistics (execution time, frame size, buffer queue size, etc.) are sent to the decision module, which keeps collecting statistics until the GOP size threshold is reached. *Line* 6 predicts the motion level for the next GOP based on the amount and change rate of recent GOPs. *Line* 7 calculates the difference between the averaged encoding time of current GOP frames and the standard encoding time (the inverse of the frame rate), which is used for motion level threshold adaptation in lines 9 and 10. *Line* 8 computes a similar value for the decoder. Based on the current threshold and predicted motion level, a proper coding set is assigned to the encoder and the decoder for the next GOP, as lines 11 to 22 show. We use adaptive instead of fixed thresholds to maintain a fair use of computational resources in the system.

Algorithm 1 Workflow of Decision Module

```
1: for each frame do
2:   CollectStats(Encoder,Decoder);
3:   if !GOP_Size_Reached then
4:     continue;
5:   end if
6:   $MotionL = \alpha \times prevMotionL + \beta \times (MotionL - prevMotionL)$;
7:   $\Delta EncTime = \overline{EncTime} - StdEncTime$;
8:   $\Delta DecTime = \overline{DecTime} - StdDecTime$;
9:   $Thresh_L = Thresh_L + \lambda \times \Delta EncTime$;
10:  $Thresh_H = Thresh_H + \lambda \times \Delta EncTime$;
11:  if $MotionL < Thresh_L$ then
12:    SetEncoderSet(L3);
13:  else if $MotionL > Thresh_H$ then
14:    SetEncoderSet(L1);
15:  else
16:    SetEncoderSet(L2);
17:  end if
18:  if $\Delta DecTime/StdDecTime > \theta$ then
19:    SetDecoderSet(D1);
20:  else
21:    SetDecoderSet(D0);
22:  end if
23:  FrameDropHardwareControl();
24:  ResetStats(Encoder,Decoder);
25: end for
```

To optimize network usage, the NIC is set to sleep mode when the buffer is emptied. When frames of a GOP have been encoded, the NIC is transitioned into working mode to send and receive data. The frame drop rate is proportional to buffer occupancy rate BOR and DVFS. The CPU frequency will be adjusted to the next lower or higher level if BOR is below or higher than some thresholds. If the highest frequency level has been reached in the previous round and BOR still remains high, more frames will be dropped and the coding set will be further lowered.

6. EXPERIMENTAL EVALUATION

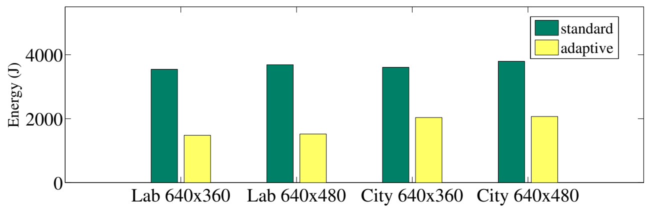
We established a video call connection between a server and a ThinkPad laptop through a WiFi network, which was measured to have sufficient bandwidth and a negligible loss rate. The laptop was equipped with a 2.8GHz Intel$^{®}$Core 2

Vid.	Res.	App	EDrop [%]	NPSNR [dB]	EQT [ms]
lab	640x360	S	0	33.992	1.24
		A	0	34.024	2.12
	640x480	S	0	32.768	18.96
		A	0.331	32.713	18.92
city	640x360	S	0	28.571	25.04
		A	0	29.325	27.36
	640x480	S	1.466	28.281	44.04
		A	3.864	28.294	51.72

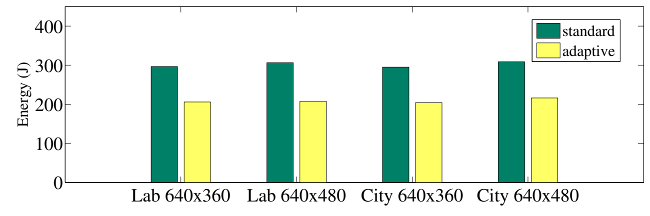
Table 3: Performance comparison at 400kbps with standard (S) and adaptive (A) approaches.

Duo processor and an Intel® Wireless WiFi Link 5300AGN NIC. The processor supports DVFS through the ACPI interface[2] at four frequency levels: 0.8GHz, 1.6GHz, 2.13GHz and 2.8GHz. The energy model for the CPU and the NIC are derived from prior work. We recorded two 2-minute-long videos in raw YUV format at resolution 640x480, then rescaled them to 640x360. The bitrate was fixed at 400kbps. For the standard approach (S), we set the frequency to the highest available value (2.8GHz) and sent every frame once it was encoded. For the adaptive approach (A), DVFS and buffered transmission are both employed. We performed our tests five times and the results were averaged. The performances are shown in Table 3. Figure 3 shows the energy consumption of the CPU and the NIC. In Table 3 *EDrop*, *NPSNR* and *EQT* stand for encoder drop rate, encoder nominal PSNR and encoder drop rate, respectively.

From Table 3 we observe that the adaptive, energy saving mode shows comparable performance to the standard mode (at the highest CPU frequency) while its energy consumption is impressively decreased. With DVFS, the CPU energy consumption is reduced by 52.04% on average, and NIC energy is reduced by 30.88% through buffered transmission.



(a) CPU Energy Consumption



(b) NIC Energy Consumption

Figure 3: Energy consumption of CPU and NIC.

7. CONCLUSIONS AND FUTURE WORK

We have proposed an integrated framework for complexity adaptation and energy conservation for mobile video calls, combining selected encoding and decoding sets for videos of different motion levels into a feedback algorithm. We also utilize hardware energy saving techniques such as DVFS for the CPU and buffered transmissions for the wireless network to reduce the energy consumption while still achieving satisfactory system performance, which is validated by our experimental results under two scenarios.

Going forward, we plan to extend and implement the feedback mechanism with a holistic decision on the execution conditions of both participants. Moreover, wireless networks can be unstable, which may result in frequent data loss and varied bandwidth. Therefore, error correction and concealment must be employed.

[2]www.acpi.info

8. ACKNOWLEDGEMENT

We would like to acknowledge that this research was carried out at the Centre of Social Media Innovations for Communities (COSMIC), sponsored and supported by the Singapore National Research Foundation and Interactive & Digital Media Program Office, MDA.

9. REFERENCES

[1] ISO/IEC 14496-2:2004 - Information technology – Coding of Audio-Visual Objects – Part 2: Visual, 2004.

[2] Z. Cao, B. Foo, L. He, and M. van der Schaar. Optimality and Improvement of Dynamic Voltage Scaling Algorithms for Multimedia Applications. In *45th Design Automation Conference*, pages 179–184, 2008.

[3] C. Hsu and M. Hefeeda. Achieving Viewing Time Scalability in Mobile Video Streaming Using Scalable Video Coding. In *1st ACM SIGMM Conference on Multimedia Systems*, pages 111–122, 2010.

[4] W. Ji, M. Chen, X. Ge, P. Li, and Y. Chen. A Perceptual Macroblock Layer Power Control for Energy Scalable Video Encoder Based on Just Noticeable Distortion Principle. *Journal of Network and Computer Applications*, 34(5), 2010.

[5] W. Ji, M. Chen, X. Ge, P. Li, and Y. Chen. ESVD: An Integrated Energy Scalable Framework for Low-power Video Decoding Systems. *EURASIP Journal on Wireless Communications and Networking, Special Issue on Multimedia Communications over Next Generation Wireless Networks*, 2010.

[6] S. Peng. Complexity Scalable Video Decoding via IDCT Data Pruning. In *IEEE International Conference on Consumer Electronics (ICCE)*, pages 74–75, 2001.

[7] M. Shafique, L. Bauer, and J. Henkel. 3-Tier Dynamically Adaptive Power-Aware Motion Estimator for H.264/AVC Video Encoding. In *13th International Symposium on Low Power Electronics and Design (ISLPED)*, pages 147–152, 2008.

[8] M. Tamai, T. Sun, K. Yasumoto, N. Shibata, and M. Ito. Energy-Aware Video Streaming with QoS Control for Portable Computing Devices. In *14th International Workshop on Network and Operating Systems Support for Digital Audio and Video*, pages 68–73, 2004.

[9] Y. Wang and S. Chang. Complexity Adaptive H.264 Encoding for Light Weight Streams. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages II25–28, 2006.

[10] W. Yuan, K. Nahrstedt, S. Adve, D. Jones, and R. Kravets. GRACE-1: Cross-layer Adaptation For Multimedia Quality and Battery Energy. *IEEE Transactions on Mobile Computing*, pages 799–815, 2006.

[11] Z. Zhou and M. Sun. Fast Macroblock Inter Mode Decision and Motion Estimation for H.264/MPEG-4 AVC. In *International Conference on Image Processing (ICIP)*, pages 789–792. IEEE, 2004.