VIDEO CODING USING COMPRESSED REFERENCE FRAMES

Madhukar Budagavi and Minhua Zhou

DSPS R&D Center, Texas Instruments Inc., Dallas, TX-75243, USA

ABSTRACT

Handheld battery-operated consumer electronics video devices such as camera phones, digital still cameras, digital camcorders, and personal media players have limited system memory bandwidth available because of cost and power consumption constraints. Video coding consumes a significant amount of this limited system memory bandwidth especially at high-definition (HD) resolution. Techniques that reduce memory bandwidth in video coding are crucial for implementing video coding at HD resolutions in portable video devices. Memory bandwidth reduction is also desirable from a power consumption point of view since memory accesses consume a significant amount of power. In this paper we present our technique - in-loop compression of reference frames - for reducing memory bandwidth in video coding.

Index Terms— Video coding, H.264/AVC, memory bandwidth reduction, low power consumption, next generation video coding standards.

1. INTRODUCTION

Handheld battery-operated consumer electronics devices such as camera phones, digital still cameras, digital camcorders, and personal media players have become very popular in recent years. Video codecs are extensively used in these devices for video capture and/or playback. The annual shipment of such devices already exceeds a hundred million units and is growing, which makes mobile battery-operated video device requirements of low-power, low-cost, and high-quality at low complexity very important to focus on in video coding research and development [1] [2]. The video coding methods developed for future video coding standards need to address the following to satisfy the requirements for video coding on mobile battery-operated video devices: memory access bandwidth reduction, enhanced parallelism support, computation cycles reduction, and memory size reduction [2]. In this paper our focus is on developing video coding techniques for reducing memory access bandwidth and also memory size.

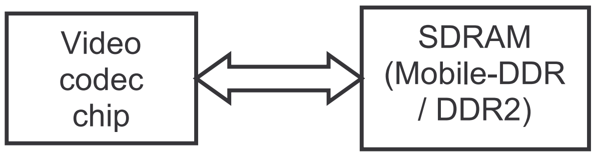

Figure 1: In portable video devices reference frames used by video codec are stored in external SDRAM.

In portable video devices, the video codec chip is connected to external SDRAM chip (which is usually either Mobile-DDR or DDR2) where video reference frames are stored. The video reference frames need to be accessed for both motion estimation and motion compensation. Portable video devices are expected to support high-definition resolution (HD) video coding in the near future. In fact, HD resolution camcorders have already started appearing in the market [3]. At HD resolution, memory bandwidth becomes one of the key limiting factors since the total available system memory bandwidth is limited in portable video devices because of cost and power reasons. Limited memory bandwidth impacts video quality because of constraints on amount of data that can be loaded for motion vector search in encoder. The power consumed for memory accesses also goes up at HD resolution because of the significant amount of data that needs to be accessed. Hence techniques for reducing memory bandwidth are critical at HD resolution.

In this paper, we present a technique for reducing memory bandwidth in video coding. Our technique consists of in-loop compression of reference frames. Since reference frame compression is carried out in the core video coding loop, there is no mismatch between the encoder and decoder. A byproduct of our technique is that the amount of memory required to store the reference frames also goes down because of reference frame compression. Note that reduced memory size is desirable in order to reduce the cost of the system. Our technique serves the dual purpose of reducing memory bandwidth and memory size.

In the next section, we consider HD video encoding and compare the power consumed for SDRAM memory accesses with power consumed in video codec chip to motivate the importance of memory access bandwidth reduction. In

Section 3, we describe our technique of video coding using reference frame compression. In Section 4, we present the rate distortion impact when H.264 is modified to include reference frame compression. We conclude the paper in Section 5.

2. SDRAM ACCESS POWER CONSUMPTION IN VIDEO ENCODING

SDRAM memory accesses typically consume around 20-30 mW/Gbps of data transfer [4] or more. Power consumed in SDRAM is given by [5]:

Total Power = Core Power + I/O Power
= (IDD4 x VDD)+(C x f/2 x VDDQ^2 x number of I/Os/2)

For example, the Samsung K4X51323PC Mobile-DDR SDRAM [6] has the following specifications at 133 MHz:

f = 133*(10^6);	% 133 MHz
Vdd = 1.8;Vddq = 1.8;	% 1.8 Volts
numIO = 32;	% Number of I/O pins
Idd4r = 125*(10^{-3});	% Read current – 125 mA
Idd4w = 100*(10^{-3});	% Write current – 100 mA

Assuming a 4 pF capacitive load (C = 4*10^{-12} in the above equations), the Samsung K4X51323PC consumes roughly an estimated 29.6 mW/Gbps for reads and an estimated 24.4 mW/Gbps for writes. Another commonly used SDRAM in portable video devices is the DDR2 SDRAM. The power consumption in DDR2 is higher than that in Mobile-DDR SDRAM (but DDR2 is cheaper than Mobile-DDR). For example, the Samsung K4T51163QE DDR2 SDRAM [7] at 200 MHz consumes roughly an estimated 41.2 mW/Gbps for reads and an estimated 31.4 mW/Gbps for writes.

Let us consider the example of HD video encoding to illustrate the total amount of power consumed by SDRAM memory accesses. The power consumed by memory accesses is the sum of core power of the SDRAM chip, the IO power of the SDRAM chip and the IO power of the video codec chip. The estimated SDRAM memory access bandwidth for motion estimation and motion compensation in a 720p30 H.264 encoder is around 380 MB/s assuming number of reference frames = 2, motion vector search range of (64 Horizontal, 32 Vertical) and a sliding motion vector search window. Note that the sliding motion vector search window algorithm maximizes the caching of motion vector search data between adjacent macroblocks. The memory access bandwidth will be much higher if caching is not used. The estimated power consumed for memory transfers in motion estimation and motion compensation in a 720p30 H.264 encoder is 98.34 mW when Mobile-DDR is used and 131.9 mW when DDR2 is used.

To illustrate the amount of power consumed in a video codec chip, consider the HDTV 720p encoder chip presented in [8]. The authors quote a power consumption number of 785 mW for their chip which is fabricated using 180nm technology. If the same chip were to be manufactured with a newer semiconductor technology such as 65nm, the power consumption of the chip presented in [8] can be expected to be roughly 170 mW if no other power optimization is done. This is because power consumption typically goes down by 40% with each shrink of transistor size. Moving from 180nm to today’s 65nm involves three shrinks of transistor size (180nm -> 130nm -> 90 nm -> 65nm) resulting in a 78% reduction in power consumption in going from 180nm to 65 nm.

When we compare the power consumed in a video codec chip with that consumed for SDRAM memory accesses based on the rough calculation presented in this section, we can see that estimated power consumed by SDRAM memory access in motion estimation and motion compensation in a video encoder is significant when compared to video codec core power. Hence, techniques that reduce memory access bandwidth are very important in reducing power in a total video encoder solution that consists of the codec core + SDRAM memory. Note that the main goal of this section was to highlight the importance of reducing memory transfer bandwidth. The calculations shown in this section are approximate and are not measured numbers.

3. VIDEO CODING USING COMPRESSED REFERENCE FRAMES

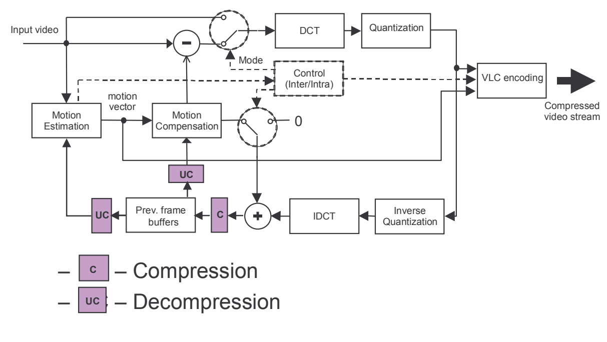

Figure 2: Video coding using compressed frame buffers.

Figure 2 shows the key idea behind our technique for reducing memory access bandwidth and reference frame buffer memory size. We compress the reference frames before storing them in memory thereby saving write memory bandwidth and we decompress them after reading them from memory thereby saving read memory bandwidth. We carry out reference frame compression in the core video coding loop so that quantization errors encountered during reference frame compression show up in the residual after motion compensation and get compensated for in both the encoder and decoder. Including reference frame compression in the core video coding loop is the key idea of our technique. It prevents the drift between the encoder and the decoder leading to a perfect match between the encoder and the decoder.

Several algorithms can be used to compress reference frames. The desirable characteristics for the reference frame compressor are:

1. Simple encoding and decoding algorithms – The encoding and decoding must not be complex as the power incurred in encoding and decoding should not offset the power savings because of the resulting memory bandwidth compression.
2. Fixed length compression for random access of memory blocks – Random access is desirable in both the encoder and the decoder to access any block of video data in the frame for motion estimation and motion compensation.

The particular algorithm that we use for compressing reference frames in this paper is a fixed-length compression algorithm based on block scalar quantization [9]. Note that other fixed-length compression algorithms such as vector quantization can be used too and is a topic for future research. Figure 3 shows the pseudo-code of the algorithm that we use (henceforth termed as min-max-scalar-quantization (MMSQ) scheme).

MMSQ encode

```
compress_block(inblk[4][4]) {
    // calculate min and max value of pixels in inblk
    min = minimum(inblk);
    max = maximum(inblk);

    // Store min and max values, N bits each
    putbits(min,N);
    putbits(max,N);

    // Scalar quantize all pixels to L bits
    Q = (mmax-mmin)/(2^L-1);
    for(i = 0 to 3, j = 0 to 3) {
        dq = floor((inblk[i][j]-mmin)/Q+0.5);
        putbits(dq, L);
    }
}
```

Figure 3: Min-max-scalar-quantization scheme (MMSQ).

The input (inblk) to the compressor is a block of 4x4 pixels, which in general can be from N-bit video signal. In the first step of the algorithm, the minimum and maximum pixel values in the block are calculated and stored. Then all the pixels in the 4x4 block are uniformly quantized to L bits between the calculated minimum and maximum pixel values and stored. The total number of input bits into the algorithm is 16*N. The total number of compressed bits is: 2*N+16*L.

The complexity of the MMSQ scheme is very low. The scalar quantization operation can be simplified as:

$$dq=floor(inblk[i][j]-mmin)*(2^L-1)/(mmax-mmin)+0.5);$$

We implement the division operation above as a multiplication by inverse. We pre-calculate the inverses and store it in a lookup table. For 8-bit video signal, the range of (mmax-mmin) is limited to [0 255], hence a table of 256 elements is sufficient.

4. RESULTS

We incorporated the MMSQ scheme into JVT JM 11.0 software to evaluate the rate-distortion performance of using compressed reference frames in H.264. The encoding options used were: IPPP-coding, rate-distortion enabled, single reference frame, JM fast full-search ME. The QP values used to calculate the Bjontegaard ΔBitrate and ΔPSNR [10] were: 22, 27, 32, 37.

Table 1: Rate-distortion performance of 6bpp MMSQ.

MMSQ 6bpp (L=5)

Sequence	ΔBitrate (BD delta)	ΔPSNR (BD delta)	Estimated bandwidth savings	Reference memory size savings
football_p704x480.yuv	0.22%	-0.01	25%	25%
ICE_704x576.yuv	1.05%	-0.04	25%	25%
mobile_p704x480.yuv	2.14%	-0.104	25%	25%
SOCCER_704x576.yuv	0.57%	-0.023	25%	25%
CREW_704x576.yuv	0.28%	-0.01	25%	25%
HARBOUR_704x576.yuv	1.61%	-0.065	25%	25%
tennis_p704x480.yuv	1.36%	-0.05	25%	25%
Average	1.03%	-0.043	25%	25%

Table 1 lists the Bjontegaard ΔBitrate increase and ΔPSNR degradation when 6-bpp MMSQ is used for compressing reference frames in H.264 when compared to H.264 with uncompressed reference frames. For representing each 4x4 block, 8-bits are used for minimum pixel value (per block), 8-bits are used for maximum pixel value (per block), pixels in the block are uniformly quantized to lie in the [minimum, maximum] range by using 5 bits per pixel (L=5). So overall, to represent a 4x4 block, we require 6 bits/pixel. This leads to a 25% savings in memory size used to store reference frames and an estimated 25% savings in memory transfer bandwidth over H.264. These significant savings come at the cost of very little degradation in rate-distortion performance as can be seen from Table 1 where the average bit-rate increase is 1.03% or equivalently the average PSNR degradation is -0.04 dB.

Table 2: Rate-distortion performance of 5bpp MMSQ.

MMSQ 5bpp (L=4)

Sequence	ΔBitrate (BD delta)	ΔPSNR (BD delta)	Estimated bandwidth savings	Reference memory size savings
football_p704x480.yuv	0.66%	-0.027	37.5%	37.5%
ICE_704x576.yuv	3.85%	-0.146	37.5%	37.5%
mobile_p704x480.yuv	7.79%	-0.368	37.5%	37.5%
SOCCER_704x576.yuv	2.09%	-0.086	37.5%	37.5%
CREW_704x576.yuv	1.32%	-0.045	37.5%	37.5%
HARBOUR_704x576.yuv	6.63%	-0.26	37.5%	37.5%
tennis_p704x480.yuv	5.12%	-0.181	37.5%	37.5%
Average	3.92%	-0.159	37.5%	37.5%

Table 2 lists the Bjontegaard ΔBitrate increase and ΔPSNR degradation when 5-bpp (L=4) MMSQ is used for compressing reference frames in H.264 when compared to H.264 with uncompressed reference frames. This leads to a 37.5% savings in memory size used to store reference frames and an estimated 37.5% savings in memory transfer bandwidth over H.264. However, these significant savings come at the cost of degradation in rate-distortion performance of 3.92% increase in average bit-rate or equivalently 0.159 dB degradation in PSNR.

5. CONCLUSIONS

Low power consumption and low-cost implementation are important requirements for video codecs used in handheld battery-operated consumer electronics devices such as camera phones, digital still cameras, digital camcorders, and personal media players. These video devices will be expected to support high-definition resolution (HD) video coding in the near future. At HD resolution, memory bandwidth reduction is essential for reducing power consumption and cost in portable video devices. In this paper, we presented a simple technique that reduces memory bandwidth and also memory size by compressing reference frames before storing them in memory. The key idea in our technique is that the reference frame buffer compression is carried out in the core video coding loop so that there is no mismatch between the encoder and decoder. Since the core video coding loop is modified, including support for our technique requires a modification to the video coding standard and it is a topic for future video coding standardization [11].

In simulations, one configuration of our technique using block scalar quantization achieves a savings of 25% in reference frame memory size and an estimated savings of 25% in bandwidth at the cost of 1% average increase in bit-rate or equivalently a decrease in average PSNR of 0.04dB on 7 D1 resolution video clips when using H.264. Other configurations of our technique allow for a trade-off in memory and rate-distortion performance.

REFERENCES

[1] M. Budagavi and M. Zhou, “Next Generation Video Coding for Mobile Applications: Industry Requirements and Technologies,” Proc. SPIE Visual Communications and Image Processing (VCIP), San Jose, Jan. 2007.
[2] Texas Instruments, Nokia, Polycom, Samsung AIT, Tandberg, “Desired features in future video coding standards,” Document T05-SG16-C-0215, ITU-T SG 16 contribution, Geneva, June 2007.
[3] Sony HDR-HC5 High Definition Handycam® Camcorder, www.sony.com.
[4] M. Horowitz, et. al., “Scaling, Power, and the future of CMOS,” IEEE Intl. Electron Devices Meeting Technical Digest, 2005.
[5] Micron Technical note TN-48-10, “Mobile SDRAM power-saving features”, www.micron.com.
[6] Samsung K4X51323PC 16Mx32 Mobile-DDR SDRAM, www.samsung.com.
[7] Samsung K4T51163QE 32Mx16 DDR2 SDRAM, www.samsung.com.
[8] T-C. Chen, et. al., ”Analysis and architecture design of an HTDV 720p 30 frames/s H.264/AVC encoder”, IEEE Trans. Circuits Syst. Video Technol., vol. 16, No. 6, pp. 673-688, June 2006.
[9] K. Sayood, Introduction to Data Compression, Morgan Kaufmann Series in Multimedia Information and Systems.
[10] G. Bjontegaard, “Calculation of Average PSNR Differences between RD curves”, ITU-T SG16/Q6, 13th VCEG Meeting, Austin, Texas, USA, April 2001, Doc. VCEG-M33.
[11] M. Budagavi and M. Zhou, “Video coding using compressed reference frames”, ITU-T SG16/Q6, 31st VCEG Meeting, Marrakech, Morocco, January 2007, Doc. VCEG-AE19.