LRU strategy for I frame reduction

Ruiduo Yang and Michael S.Brown

Department of Computer Science
Hong Kong University of Science and Technology
Clear Water Bay, Kowloon, Hong Kong
{yangrd,brown}@cs.ust.hk

Abstract

Frequent placement of intra-encoded pictures, or I-frames, in MPEG video facilitates (1) error resilience over lossy network transmission and (2) random access for VCR like functionality. However, high I-frame frequency sacrifices quality-to-bitrate efficiency that can be gained by using longer sequences of inter-encode pictures. In this paper, we present a simple strategy that emulates frequent I-frame encoding while using fewer I-frames. Our approach maintains small set of previously encoded/decoded I-frames that can be re-used to start future GOPs. We overview our approach and show how a least-recently-used (LRU) policy can be used to maintain the set of I-frames. We demonstrate gains in PSNR for constant bit rate encoding using our strategy.

1. Introduction

MPEG-based encoded video uses a combination of intra-encoded and inter-encoded frames to compose its compressed video stream. Intra-encoded frames (I-frames) can be independently encoded and decoded. Inter-encoded frames (P-frames and B-frames) exploit temporal redundancy using motion compensated residual coding strategies. P/B frames encode their difference to one or two reference frames and are dependent on these reference frames for their reconstruction. MPEG organizes encoded frames into a structure called a group of pictures (GOP), which starts with an I-frame followed a series of inter-encoded frames. The size of the GOP can be considered the distance between I-frames. In terms of bits-per-frame compression, I-frames are often several times larger than inter-encoded frames. Thus, higher quality-to-bitrate can typically be achieved by using long sequences of inter-encode frames.

The MPEG encoding syntax allows encoder decisions to be made that can affect quality-to-bitrate performance. This has lead many researches to explore improvements by adaptive I-frame placement, or dynamic GOPs. For example, Lan et al[1] used scene motion detection to determine content change for I-frame placement, using a long run of inter-coded frames until a substantial scene change was detected. Yoneyama et al[7] examined macroblock motion vector activity to determine where to start new GOPS. Turaga et al [4] presented as classification approach, training a video-encoder with sample video clips to guide the I-frame selection. While all of these approaches reported quality-to-bitrate improvements, their use of arbitrary sized and typically long GOPs has some undesirable drawbacks over high I-frame placement.

Frequent I-frames placement provides two important functions. First, because I-frames are self-contained, they provide a mechanism to randomly access frames in the compressed stream. Random access allows applications to provide VCR like functionality such as fast-forward and rewind. I-frame frequency determines the granularity of this random access. Second, I-frames provide error resilience for noisy network transmission. If a portion of a transmitted frame is corrupted, the error can be propagated by the predictive nature of B/P frames. Sending frequent I-frames can help stop such pixel error propagation.

The challenge then is to provide the functionality of frequent I-frame placement while providing high quality-to-bitrate efficiency. We address this problem using an encoding strategy that reduces the number of I-frames while maintaining the functionality of frequent I-frame placement. Our approach maintains a small set of previously en/decoded I-frames, called the working-set. Working-set frames can be used as reference frames of the start of future GOPs often avoiding the encoding of new I-frames. Moreover, each GOP's start frame can still be randomly accessed because the necessary reference frame is buffered in the working-set. In the remainder of this paper, we describe in detail our approach (section 2) and show that a least-recently-used strategy is effective for maintaining the working-set frames (section 3). Section 4 explains our integration into an MPEG-2 codec

and demonstrates our results. We finish with concluding remarks in section 5.

2. Overview of the working-set frame replacement

We borrowed the term ``working-set'' from operating system's vernacular which describes a memory paging approach for virtual memory. In an OS context, a process keeps a set of memory pages (called its ``working-set'') that are pre-fetched onto the system's memory when the process is context switched onto the CPU. The working-set is a reasonable-guess of the most useful pages (useful in terms of memory access hit ratios) for the process. We liken our I-frame replacement problem to memory paging. In a video coding context, our ``working-set'' is a reasonable-guess of the most useful I-frames that have already been encoded. These I-frames are useful in that they are similar enough to future I-frames to be used as reference frames for subsequent I-frames.

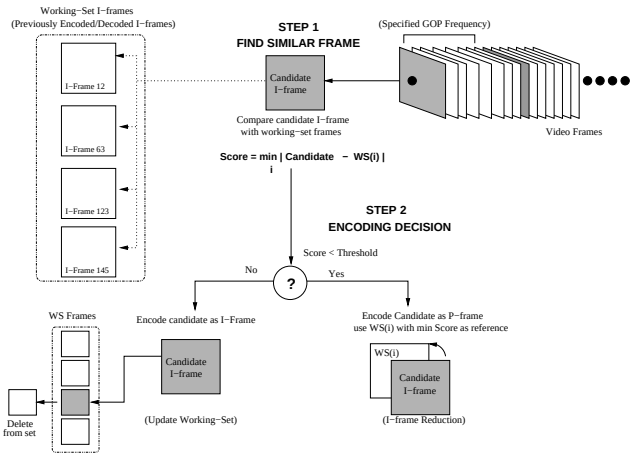

Figure 1: Overview of the working-set frame replacement. Both the encoder and decoder keep a set of previously encoded/decoded I-frames. The I-frame frequency is specified by a GOP size. When it is time to encode a new I-frame (called a candidate I-frames), the most similar working-set frame is found using an SAD metric. If the most similar frame passes an ``acceptance threshold'', the candidate frame is inter-encoded using the similar working-set frame as a reference. Otherwise, the current frame is encoded as an I-frame and is placed in the working-set using an LRU replacement policy.}

Figure 1 overviews of the working-set replacement strategy. A small set (for example 4 frames) of previously coded I-frames are maintained on both the encoder and decoder. Frames are considered for intra-frame encoding based on a specified placement frequency, i.e. fixed GOP size. In our experiments, we use a GOP size of 12, a reasonable granularity for rewind and fast-forward functionality. When a new GOP is to be started, a decision is made whether or not to encode the frame as an I-frame. In figure 1, we refer to this frame as a candidate I-frame. The candidate I-frame is compare against all the frames in the working-set (step 1). We use a simple sum-of-the-absolute-difference (SAD) as follows:

$$|SAD| = \min_{i \in k} \sum_{x,y}^{w,h} |WS_i(x, y, mv_{x,y}) \quad I_{candidate}(x, y)|$$

In the above equation, i represents an index for each k working-set frame. $mv_{x,y}$ is the motion vector indicating the displacement between the candidate frame and WS_i at point (x, y), this motion vector can be acquired using a block based motion estimation step. A pixel-wise difference is performed for pixel (x, y) over the width w and height h of the frame's luminance channel.

The minimum SAD score is examined to see if it is less than a specified acceptance threshold, i.e. $Score < T_{ws}$. To make specifying the threshold easier, we normalize the SAD score by dividing it by the number of pixels in the frame -- this results in a score with a range of 0-255 (the pixel intensity range). If the minimum SAD score is less than the threshold, the candidate I-frame will be encoded as a P-frame, using the most similar WS_i frame as a reference. In this manner, the GOP does not start with an I-frame, but instead starts with a P-frame that uses a buffered working-set frame as a reference. Random access can still be achieved using this approach. The start of each GOP is either an I-frame or is P-frame whose reference is buffered in the working-set.

We note that the idea of buffering many frames on the encoder and decoder is not new. Weigand et al [5,6] introduced the idea of multiple reference frames (called long term memory frames) for use in inter-frame encoding. In their strategy, inter-frame's macroblocks can predict themselves from any of the buffered reference frames. This strategy has been adopted in H.263++, annex U. Using multiple references for inter-encoding increases coding complexity, but has been shown to provide significant quality-to-bitrate improvements. Our idea is in the spirit of Weigand et al, but differs in several distinct ways. For example, we only maintain I-frames in the working-set. In addition, our approach is only targeting the start of each GOP and not all inter-encoded frames. We also use a global similarity check to find a suitable reference frame and not at the macro-block layer. Finally, our scheme is intended for little computational overhead to the encoder. In this section, we proposed a method to do error

3. Lru working set maintenance

To maintain the working-set frames, we again turn our attention to virtual memory approaches, treating frame replacement as a paging problem. There are several schemes for memory paging, including first in first out (FIFO), not used recently (NUR), least recently used

(LRU) and least frequently used (LFU) (see [2] for a refresher on memory paging). It is known that LRU is one of the most effective paging algorithms; however, due to implementation overhead it is rarely deployed in a virtual memory context [2]. However, for our problem LRU is suitable for deployment.

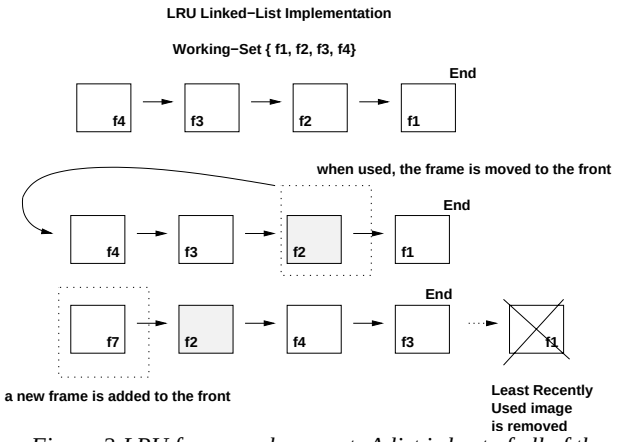

Figure 2:LRU frame replacement: A list is kept of all of the frames. When a new frame is added, or a frame in the working-set is chosen to replace a candidate I-frame, it is moved to the front of the list. The least recently used frame is always the at end of the list.

As the name implies, LRU replaces the frame (or page) that was least recently used. The LRU implementation uses a fixed-sized linked-list data structure to store indexes for all of the frames, as shown in figure 2. Newly encoded I-frame is added to the front of the linked list. When a frame in the list is used as a reference (i.e., replaces a candidate I-frame), it is moved to the front of the list. With this strategy, we see that the frame at the end of the linked list is always the frame that has been least recently used. When the list is full and a new frame is to be inserted, we simply remove (delete) the frame at the end of the list, and place the new frame at the beginning.

For virtual memory systems, LRU requires at each memory access, the appropriate page is removed from a linked-list and placed at the beginning. This per-access overhead is impractical in a virtual memory context. However, with our framework LRU is quite feasible; assuming a maximum of 30 video frames-per-second, we would only need to perform SAD scores and LRU update a few times a second for short GOP patterns. Real-time performance can be easily realized.

4. simulation results

4.1 Integration to TM5

We have integrated our proposed approach into an MPEG-II encoder codec [3]. We changed the grammar of MPEG bitstream slightly to add extra bits to convey working-set update information. MPEG-II pictures start with a *Pict_type* field, specifying either I,B, or P frame. If *Pict_type* specifies an I-frame, we add 32 bits as follows:

PICT_TYPE	WS_TYPE**(1 BIT)**	REF_NUMBER**(31 BITS)**

If encoded frame is an intra-encoded frame, the *ws_type* field is set to 0. The decoded I-frame is added to the working-set. If the following 31 bits are set to 0, then no I-frame will be replaced (as in the case when the working-set is not full). Otherwise the 31 bit *ref_number* specifies which frame to replace in the buffer.

If the frame is inter-encoded, the *ws_type* bit is set to 1 and the following 31 bits indicate the index of the frame to be used as a reference. In our current implementation we assume that I-frames will not be lost and the reference number is consistence on both the encoder and decoder. For network transmission, a feedback mechanism will be needed to inform the encoder which I-frames have successfully received. This is currently deferred to future work.

On the decoder side, by examining the 32 bits following an I-frame *Pict_type* field, the decoder can maintain the exact working-set as used by the encoder. For random access, the implementation of fast-forward is easy to realize. For rewind, we can examine the 31 bits to fetch discarded frames. It could be that the fetch frame is temporally far from the current frame. However, if this is the case, it implies that the fetched frame stayed in the working-set a long time and will be referenced often, thus its fetch time is offset by its usefulness.

4.2 Results

We compare our modified encoder to the baseline MPEG-II encoder. We compare the PSNR of our method using the following acceptance thresholds: 5, 10, and 15. At this stage, we manually choose the working-set threshold. We are examining techniques to automatically set these values.

We show results for two sequences, Paris and News. We use the following GOP IBBPBBPBBPBB of size 12. During motion estimation, motion vector range is chosen to be [-16, +15.5], [-8, +7.5] for P frames and B frames with halfpel accuracy respectively. MQUANT value is estimated by the same rate control method in TM5 to obtain the specified bitrate, given in terms of bits-per-pixel (bpp). No interlaced frame is used. We use a working-set size of two frames.

Method (# of I – frames)	0.1bpp	0.2bpp	0.2bpp	0.4bpp	0.5bpp
TM 5	89	89	89	89	89
Thresh=5	82	81	78	78	76
Thresh=10	21	20	18	17	17
Thresh=15	6	5	5	4	4

Table1: Number of intra-encoded frames for different bit-perpixel (bpp) using the base-line MPEG codec and our modified codec using different similarity thresholds.

Table 1 shows the number of I-frames actually encoded using the different thresholds for the paris sequence. The number of I-frames changes slightly depending on the bpp specified. This is because the SAD scores will yield different results based reconstructed frame.

		TM5	Thresh=5	Thresh=10	Thresh=15
Akiyo	0.1bpp	37.59	39.37	39.37	39.37
	0.2bpp	41.92	42.94	42.94	42.94
Brideg-Far	0.1bpp	30.52	31.63	31.63	31.63
	0.2bpp	32.64	33.40	33.42	33.42
Container	0.1bpp	34.30	35.66	35.54	35.37
	0.2bpp	36.51	37.80	37.63	37.47
Foreman	0.1bpp	30.25	30.33	30.96	30.90
	0.2bpp	34.04	34.34	34.57	34.49
News	0.1bpp	35.90	37.48	37.44	37.44
	0.2bpp	38.50	40.10	39.92	39.92
Paris	0.1bpp	25.80	25.80	27.03	27.38
	0.2bpp	29.70	29.70	32.10	31.69

Table 2: Comparison between TM5 and different threshold settings of TM5 with LRU method based on the Average PSNR value of Y component. Workingset size is set to be 2.

Table 2 show that by reducing I-frames encoding, gains are made to the the average PSNR of both sequences by around 1.5 – 2 dBs with the threshold=10-15. Figure 3 shows a frame-by-frame comparison of PSNR for our method and the baseline TM5. We see that the majority of frames have PSNR improvements.

5. Conclusion

In this paper, a new strategy for reducing the number I-frames while emulating the placement of frequent I-frame is proposed. Our approach maintains a working-set of previously coded I-frames that can be re-used to start future GOPs. By re-using old I-frames to start new GOPs we reduce the total number of I-frames; this translates into an improvement in PSNR for constant-bit-rate encoding. We show that a least-recently-used replacement policy is suitable for maintaining the working-set, and discuss how to modify a MPEG codec to integrate the strategy.

6.Reference

[1] A. Y. Lan, A. G. Nguyeng, and J. N. Hwang. “Scene-content-dependent reference frame placement for mpeg video coding.”, IEEE Trans. Circuits and Systems Video Technology, 9(3), 1999.

[2] A. Tanenbaum. Modern Operating Systems, chapter Memory Management – Page Replacement Algorithms, pages 214–228. Prentice Hall, third edition, 2001.

[3] Test Model 5. ISO/IEC JTC1/SC29/WG11 N2459, 1994.

[4] D. Turaga and T. Chen. “I/p frame selection using classification based mode decision.”, In IEEE International Conferencing on Image Processing, Thessaloniki, Greece, October 2001.

[5] T. Wiegand, E. Steinbach, A. Stensruf, and B. Girod. “Multiple reference picture video coding using polynomial motion models.”, In IEEE Visual Communication and Image Processing, 1996.

[6] T. Wiegand, X.Z. Zhang, and B. Girod. “Long-term memory motioncompensated predicition.”, IEEE Trans on Circuits and Systems for Video Technology, 9(1), Feb 1999.

[7] A. Yoneyama, Y. Nakajima, H. Yanagihara, and M. Sugano. “MPEG encoding algorithm with scene adaptive dynamic GOP structure.”, In IEEE Third International Workshop on Multimedia Signal Processing, pages 297–302, 1997.

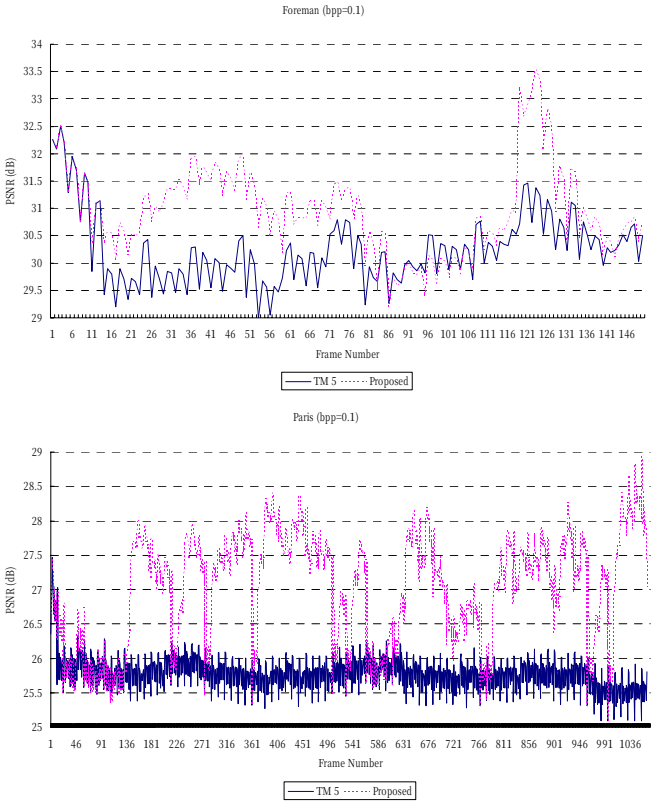

Figure 3: Comparison between TM5 and TM5 with LRU method based on the PSNR value of Y component of each frame for the Paris and Foreman sequencee. (bitrate = 0.1bpp, workingset size = 2, threshold = 10.)