A NOVEL FRAME SKIPPING METHOD IN TRANSCODER, WITH MOTION INFORMATION, BUFFER FULLNESS AND SCENE CHANGE CONSIDERATION

Sourya Bhattacharyya Emiliano Piccinelli[#]*

*Advanced System & Technology, STMicroelectronics Pvt. Ltd., Greater Noida, U.P. 201308, India,
[#]Advanced System & Technology, STMicroelectronics Pvt. Ltd., 20041, Agrate Brianza (MI), Italy,
{sourya.bhattacharyya, emiliano.piccinelli}@.st.com
web: www.st.com

ABSTRACT

Frame-skipping is basis for reducing output bit rate in real time transcoding over bandwidth-constrained channels, keeping buffer conditions under control. One approach consists in skipping frames whenever encoder buffer is going to overflow: the drawback is that high motion and scene change frames too might be skipped, incurring in visual artifacts and average compression reduction. Another method is to track motion changes occurrences and to skip low motion complex frames, to maximize output quality of encoded high motion frames. Nevertheless, in low bandwidth channels, high motion frames encoding requires more bits than average. So this approach does not always conform to buffer compliancy. In this paper, we propose a combination of these two fundamental approaches. It monitors buffer fullness in frame by frame manner and also concerns about quality encoding of scene change and high motion frames. Experimental results show that it can produce high quality output even in very low target bit rate transcoding.

1. RATE CONTROL IN TRANSCODING

MPEG2 [1] is the current wide established digital video transmission standard. H.264/AVC [2] advanced video coding standard is getting popularity for its high compression efficiency and transmission unit based error resilient mechanisms. Conversion between these two video standards with changes in bit rate, syntax, and resolution is done by video transcoding. STMicroelectronics proprietary transcoding library DBS (Dynamic Bitstream Shaper) [3] implemented MPEG2 to H.264 transcoder architecture is shown in figure 1. It uses decoded yuv image, macroblock (MB) and motion vector (MV) information via shared memory structure and uses those in encoding part, thus completely avoiding motion re estimation phase, achieving almost same output quality in real time environment.

Constant bit rate control (CBR) technique, employed in transcoding, achieves constant output bit rate by varying quantization parameter (QP) according to image complexity (X). If X increases, then QP is also increased to maintain constant output bit rate; decreasing X means lower QP. Thus output quality varies according to QP variation.

VBV (Video Buffer Verifier) [1] or HRD (Hypothetical Reference Decoder) [2] specifications replicate real time bandwidth constrained environments in video coding standards. In essence, VBV/HRD simulates decoder behavior; VBV/HRD buffer fullness is decremented when encoder generates bit streams and decoder withdraws those bits from receiver buffer. Similarly, buffer fullness is incremented as per average input bit rate fed in decoder.

We can represent VBV/HRD buffer fullness update in frame by frame manner by following formula:

$$W_{t+1} = W_t - B + R/F. \quad (1)$$

Where, W_t = HRD buffer fullness at time unit t, B = bit used in coding last frame, R = target output bit rate, and F = input frame rate. For successful decoding, rate control system must ensure that VBV/HRD should never underflow or overflow, i.e. VBV/HRD buffer fullness at any instant should always be within (0, VBV/HRD buffer size) range.

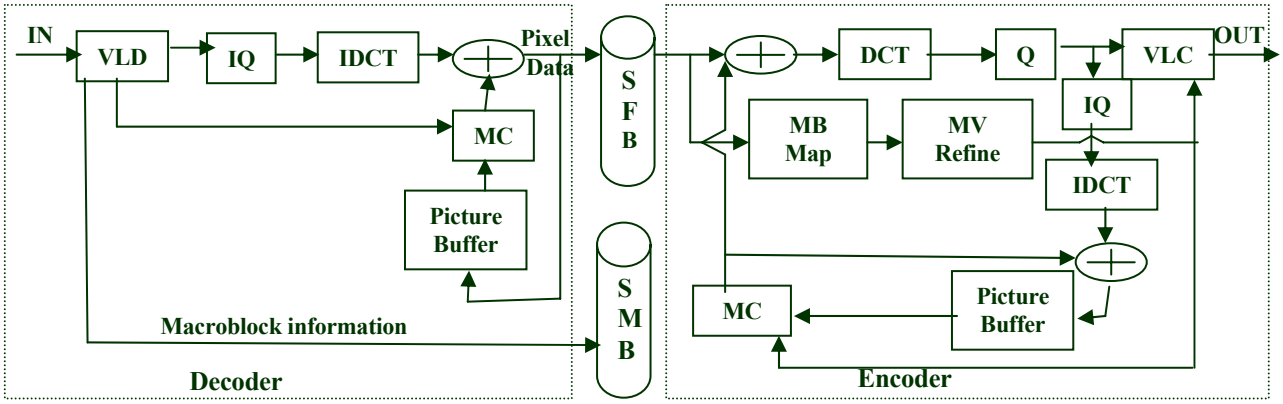

Figure 1 – Our Proposed transcoder architecture

Our DBS transcoder uses buffer fullness based single pass CBR algorithm. Rate control is function of current buffer fullness and applied QP, without any look-ahead feature. The focus is to maintain same VBV/HRD buffer fullness, before and after encoding a group of pictures (GOP), throughout the sequence. Thus every GOP is coded with almost same no of bits, generating CBR output. We use two buffer fullness based thresholds– i) overflow (brown) and ii) underflow (red). They are set within (0, buffer size) range, in a frame by frame manner, as shown in figure 2. We maintain buffer fullness within these two thresholds for every frame for ensuring buffer compliancy throughout. For a frame encode, if buffer fullness goes below red line (i.e. encoder is producing more bits than average) then QP is increased. Similarly if buffer fullness goes above brown line (encoder produces less number of bits) then QP is decreased. In both cases, new value of corresponding threshold is computed. We also check and update QP after encoding row of MB within a picture.

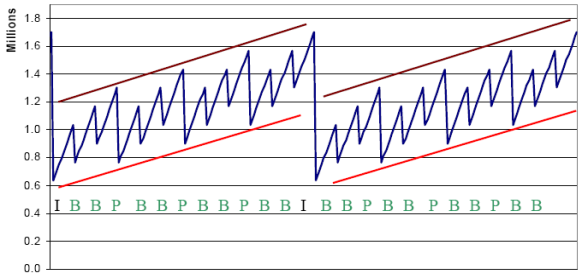

Figure 2 – HRD buffer fullness (Y axis) vs. Frame (X axis) graph.

Detection threshold values are function of i) Input bit rate and frame rate, ii) Last picture bit consumption, iii) Picture type and position with respect to current GOP. For higher input bit rate (more buffer size), thresholds values are larger, providing more bits per picture coding. For intra picture or starting pictures within current GOP, bit consumption is high and thus thresholds are lowered.

2. FRAME SKIP BASICS

In low input bit rate, without sole QP decrease mechanism alone, we skip some low complex frames so as to maintain buffer compliancy, do quality encoding of other complex frames, and produce overall good output video quality without jerkiness. We cannot employ highest QP throughout the sequence, or skip maximum of input frames because both leads to poor output quality.

2.1 Preserving scene change frames

A scene change frame [4] is a P frame having large number of Intra coded macroblock or a B frame having large number of backward reference motion vectors.

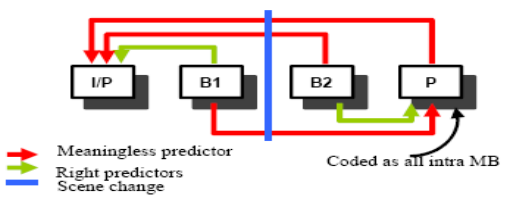

Figure 3 – Scene change occurrence (display order)

Figure 3 shows B2 as a scene change frame. Due to sudden change of content, scene change frames should not be skipped unless buffer fullness is very low.

2.2 Preserving high motion frames

A frame's motion activity (MA_{frame}) is basically average of motion vectors of its' inter macroblocks.

$$MA_{frame} = (1 / n) * \sum_{i=1}^{n} MV_i$$

Where, n is the no of non-intra macroblock of that frame, MV_i is the motion vector for i'th macroblock.

A frame with a high motion activity (MA_{frame} > a predefined motion threshold) needs to be encoded to prevent loss of information for current frame as well as for frames referencing this frame. Unless, output video will not conform to sudden motion changes or details (resulting jerky quality).

2.3 Skipping mostly low motion B frames

I frames at start of GOP contain zero motion information – they are content wise important. I and P frames serve as reference for future frames. So low motion complex B frames are the best candidate for skipping. Though depending upon buffer fullness, we may be forced to skip P or I frames also.

3. RELATED WORKS

There are lots of frame skip methodologies, based on analyzing motion activity, buffer fullness and spatio-temporal quality of output video sequence. For example, method in [5] presents minimal frame skip procedure with encoder having minimal look-ahead capability for analyzing future frames' complexity - not suitable for real time transcoding. Method in [6] skips frames whenever HRD buffer fullness is less than a static threshold (a static percentage of buffer size) – it can lead to frequent skip of scene change and high motion frames. Method in [7] deals with motion activity and motion continuity calculation between successive frames. It analyzes past GOP frames' motion change to calculate current frame's motion continuity – thus having much space and time complexity. Method in [8] considers motion change, buffer fullness, and QP based spatial quality for frame skip. Like [6], it uses static buffer size based thresholds to skip high motion and scene change frames. Moreover it uses large GOP size as test case, ignoring periodic I frame occurrence related QP and frame skip adjustments done in real time transcoding (where GOP size is generally 12-15). Method in [9] skips frames as long as motion change between previous coded frame and current frame exceeds a certain threshold, which is computed based on current buffer fullness and no of skipped frames. Thus sudden scene change and periodic intra frames can be skipped in a large number; resulting erroneous motion re-estimation for future frames and poor output quality. Finally method in [10] presents arbitrary frame skipping via sliding window mechanism. But the employed spatial temporal complexity analysis is very much space and time complex, not suitable for real time.

4. OUR ALGORITHM

Our approach of frame skip is a combination of analyzing buffer fullness, average picture bit consumption and QP usage. In process, we calculate each frame's motion activity and scene change existence for skip decision.

A) Buffer Fullness Review

We set different dynamic buffer fullness based thresholds to take encoding decision of next scene change, intra or P/B frames. For example, we skip next I frame only if,

$BF < (C_1 * Avg_{I\_frame})$ where BF = Current buffer fullness, Avg_{I_frame} = average bit consumption of all I frames so far, and C_1 is a constant scalar. Other thresholds based on average scene change frame bit consumption, average sub-GOP bit consumption and combination of these are used as benchmark for deciding whether current buffer fullness is very low, or towards lower side.

We can see that thresholds are dynamic – they are updated based on average bit consumption of frames in sequence. For very low buffer fullness, we forcibly skip next frame; otherwise we code high motion, scene change or intra frames and mark next low motion P/B frames for skipping.

```
Procedure Frameskip ()  {
A. if (buffer fullness < scene change frames' average bit
consumption so far)
        Skip scene change or high motion frame
B. if (buffer fullness < intra coded (I) frames' average bit
consumption so far)
        Skip intra or high motion frame
C. if (buffer fullness is less than some multiple of sub-
GOP size)        {
        Encode scene change / high motion / intra frame
        For next 1 or 2 low motion P or B frames
                TestAndSkip_P_or_B_Frame ()    }
D. if ((last Pict bit consumption > average bit per picture)
  Or (last picture coding causes picture underflow))    {
    Increase QP
    if average QP employed throughout picture is already
      Quite high, then
      For next 1 or 2 low motion P or B frames
        TestAndSkip_P_or_B_Frame ()    }
}       //end procedure FrameSkip

Procedure TestAndSkip_P_or_B_Frame () {
if (average bit consumption for P/B frame is very low
   Compared to average bit per picture)
        Then don't skip P/B frame
```

Figure 4 – Frame skip pseudo code

B) Average picture bit usage analysis

To preserve buffer bits for sudden scene change or high motion frame encoding, we mark low motion frames (preferably B frames) for skipping, whenever buffer fullness is towards lower side. Suppose we mark next B frame for skipping. Before skipping that B frame, we check whether average bit consumption of B frames so far is very low compared to average bit allocated for it. If so, then there is no gain in skipping that B frame unless buffer fullness is very low. By this process, we are able to maintain low rate of skipping those frames which originally have good prediction.

C) Past picture bit and QP usage

If some picture takes too much of bits during encoding, resulting picture underflow (i.e. buffer fullness going below pre-computed red threshold) then we analyze current buffer fullness and last picture average QP usage. If current buffer fullness is not very low and average QP usage of last frame is low then we increment QP. But if buffer fullness goes low or average QP employed is already quite high then we mark next P/B frames for skipping. Of course those pictures are skipped according to condition in (B).

D) B frame skip for skipped original references

For B frames, if its original forward and backward reference frames are skipped then we skip current B frame to save bits for next frames and also due to the fact that B frames having both references skipped can't have a good prediction.

Overall pseudo code of our approach is shown in figure 4.

5. MOTION VECTOR REFINEMENT

Frames having motion vector with respect to a skipped reference frame have to refine their MV's with respect to a previously coded non-skipped frame. We employ **forward dominant vector selection** (**FDVS**) procedure ([11] & [12]) for our MV re-calculation process.

For a typical MPEG2 input sequence shown in figure 5, for n^{th} reference (I/P frame), frames (n+1) and (n+3) are, respectively, backward and forward predicted from frame (n). Frame (n+2), if it is P-frame, is also forward predicted from frame (n). When frame (n) is dropped, from figure 6, forward references in frame (n+3) and backward references in frame (n+1) for macroblock such as MB_1 become invalid as they point to the dropped frame. In these figures, MB_1' represent best matching block to MB_1 and MB_1" represents a best matching block to MB_1'.

For frame (n+3) in Fig. 6(a), since MB_1' is not on a macroblock boundary, $V_1^{(n)}$ is not available from the incoming bitstream, hence, a vector addition of $I_1^{(n+3)}$ and $V_1^{(n)}$ to locate MB_1" is not possible. FDVS method derives new approximated motion vector of current coded frame by using forward reference of skipped reference frame. In figure 6, FDVS method selects dominant MV of MB_1 by calculating MV carried by a macroblock having largest overlap with MB_1'. Thus, by FDVS method, $V_1^{(n)}$ is approximated to be $I_2^{(n)}$ in figure 6(a). For backward predicted macroblock of frame (n+1), such as MB_1 in Fig. 6(b), we add $I_4^{(n)}$ to $I_1^{(n+1)}$ to locate MB_1" in frame (n-2) and then convert the prediction direction of MB_1 from backward to forward. For bi-directional predicted macroblock we simply convert the prediction direction to forward.

Multiple reference frames dropping can be handled by cumulatively composing the MVs and then storing the MVs of the reference frames. We need two tables to store the composed

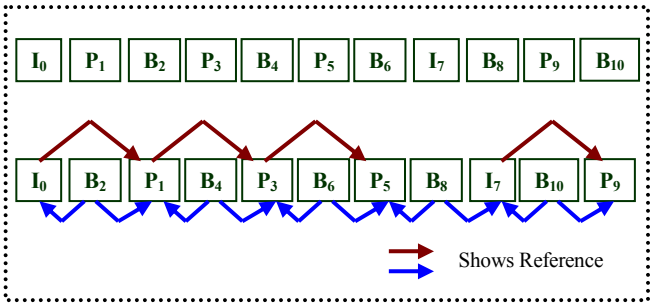

Figure 5 – MPEG2 input sequence – a) coding b) display order

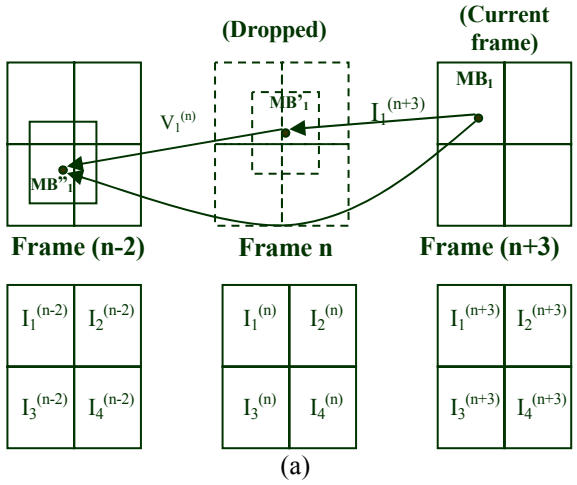

Figure 6 – Motion vector refinement for a skipped reference frame (a) Forward reference (b) backward reference

MVs corresponding to forward and backward reference frames. For example, if reference frame *i.e.* frame (n-2) for frame (n) is also dropped, then MVs for frame (n) are obtained in similar manner to frame (n+2) above and stored in a forward table. This stored MVs in the forward table are then used in the above process to compose MVs for frame (n+1), frame (n+2) and frame (n+3).

While processing frame (n+2), its composed MVs are stored in the backward table. The contents of forward and backward tables are swapped at the occurrence of next reference frame in coding order.

6. EXPERIMENTAL RESULTS

We apply our frame skip algorithm in MPEG2 to H.264 transcoder implemented in DBS [3] transcoding library. MPEG2 input QCIF sequences at 512 kbps bit rate and 25 fps frame rate is formed from standard .yuv video sequences (such as "carphone", "mobile", and "foreman") by MPEG2 reference encoder [13]. These MPEG2 sequences are transcoded to H.264 format at 64 and 32 kbps bit rate respectively. Transcoder avoids motion estimation phase by using input MPEG2 MV to calculate output H.264 compatible MV. Frame skip algorithm maintains HRD compliancy during low bit rate transcoding, as shown in figure 7.

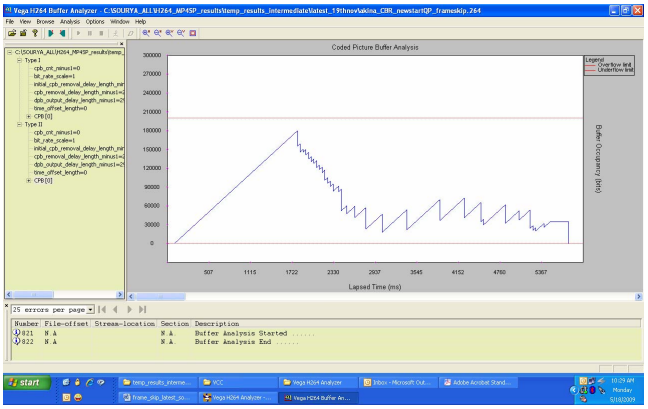

Figure 7 – HRD buffer fullness compliancy for stream "Akina" transcoded at 100 kbps, CIF resolution, and 25 fps (VEGA [17] buffer analyzer output.

We encoded same input QCIF sequences with a standard JM H.264 encoder [14] at 32 and 64 kbps with JVT rate control method [15]. We calculate frame by frame average PSNR with respect to coded or non-skipped frames only, and compare output PSNR values obtained in transcoding and encoding environments, as shown in tables 1 and 2.

From the results, our frame skip with motion refinement algorithm used in transcoding, produces better or almost similar output quality (measured in PSNR) than JVT rate control used in encoding environment, with HRD compliancy maintenance. Figure 8 shows rate distortion (R-D) curve for our algorithm for different standard QCIF sequences. In all cases, output frame by frame average PSNR values are quite consistently high, ensuring good output quality even in very low bit rate transcoding.

Our CBR along with frame skip methodology is computationally much less complex. To demonstrate this, we compared performances of our CBR algorithm (integrated in MPEG2 main to H264 baseline profile transcoder of DBS [3] library), with JVT rate control scheme (integrated in JM H.264 baseline encoder [14]). MPEG2 input CIF stream named "films" of 40 frames is fed in transcoder. Output bit rate is 512 kbps and frame rate is 25 fps. GOP size is 12. MPEG2 decoded "films.yuv" is fed in stand alone JM encoder with same output configuration. We executed transcoder and encoder in ST200 [16] micro toolset R5.1 (target architecture st231) to compare clock cycles between

two rate control schemes. Figure 9 shows the timing comparison. Our CBR is 19 times faster than JVT.

Stream Name	No of frames	Frame By Frame Avg. PSNR (dB)	
		CBR in DBS Transcoding	JVT in H.264 encoding
Bridge_close	2001	29.23	29.19
Bridge_far	2101	36.37	37.18
Coastguard	300	27.07	26
Carphone	382	28.87	26.79
Foreman	300	27.48	24.99
Hall	300	28.53	27.93
Mobile	300	21.08	18.13
Silent	300	28.2	27.3

Table 1: Frame By Frame Avg. PSNR comparison for Transcoder CBR and Encoder JVT for QCIF sequences at 32 kbps, 25 fps.

Stream Name	No of frames	Frame By Frame Avg. PSNR (dB)	
		CBR in DBS Transcoding	JVT in H.264 encoding
Bridge_close	2001	31.8	32.4
Bridge_far	2101	38.78	39.16
Mobile	300	24.1	21.3
highway	2000	33.8	35.5
Grandma	870	34.8	36.2

Table 2: Frame By Frame Avg. PSNR comparison for Transcoder CBR and Encoder JVT for QCIF sequences at 64 kbps, 25 fps.

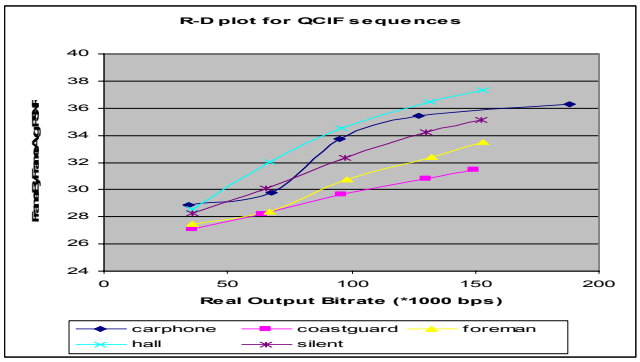

Figure 8 – R-D Plot for Our Algorithm in QCIF sequences

7. CONCLUSION

Our proposed generic buffer fullness based CBR and frame skip algorithm can be used in any transcoding environments. In main to baseline profile transcoder, we apply procedures related to B pictures in input B to output P converted pictures (with property of being used as reference). Our CBR algorithm used QP is H.264 standard based – we employ suitable QP mapping for other video format based transcoders.

REFERENCES

[1] ISO/IEC 13818-2: ITU-T H.262 Rec., 1995.

[2] ITU-T Rec. H.264 | ISO/IEC 14496-10 AVC, JVT-D157.

[3] G. Filippini, E. Piccinelli, and F. Rovati, "Method for transcoding compressed video signals, related apparatus and computer program product therefor" –Patent: EP1628484, Feb 22, 2006.

[4] S. Tripathi, and E. Piccinelli, "A Scene Change Independent High Quality Constant Bit Rate Control Algorithm for MPEG4 Simple Profile Transcoding"- IEEE Int. Symposium on Broadband Mult. System and Broadcasting, Las Vegas, NV, USA, 2008.

[5] G. Motta, J. A. Storer, and B. Carpentieri, "Frame Skipping Minimization in low bit-rate video coding", IEEE signal processing systems design and implementation, 2-4 N0V 2005, pp. 673-677.

[6] Ribas-Corbera, J., and Lei, S.: "Rate control in DCT video coding for_low-delay, communications", IEEE Trans. Circuits System Video_Technology, 1999, 9, (1), pp. 172–185.

[7] H. Shu and L.-P. Chau, "Dynamic frame-skipping transcoding with motion information considered", Image Processing, IET, December 2007, Volume 1, pp. 335-342, ISSN: 1751-9667

[8] F. Pan, X. Lin, S. Rahardja, K. P. Lim, Z. G. Li, D. J. Wu, W. Si, and L. J. Jiang, "Variable frame rate encoding via active frame-skipping", 7th International Symp. on Sig. Proc. and Its App., Vol. 1, pp. 89 – 92, July 2003.

[9] J.N.Hwang and T.D.Wu, "Motion Vector Re-estimation and Dynamic Frame-Skipping for Video Transcoding", 1998, ISBN 0-7803-5 148-7

[10] C. Y. Chen, C. T. Hsu, C. H. Yeh, and M. J. Chen, "Arbitrary frame skipping transcoding through spatial-temporal complexity analysis", IEEE Region 10 conference 2007, pp. 1-4.

[11] J.Youn, M.T.Sun, and C.W.Lin, "Motion Vector Refinement for high performance transcoding", IEEE Transaction on Multimedia, Vol 1, NO 1, March 1999

[12] V.Patil, and R.Kumar, "An Arbitrary Frame Skipping Video Transcoder", 2005, ISBN 0-7803-9332-5

[13] MPEG encoder and decoder reference software, version 1.2, 1996, MPEG software simulation group, (ftp://ftp.mpeg.org/pub/mpeg/mssg/)

[14] H.264 encoder and decoder reference software, JM version 9.6, (http://iphome.hhi.de/suehring/tml/)

[15] S. Ma, W. Gao, and Y. Lu, "Rate control in JVT standard", JVT-D030, July 2002

[16] ST200 micro toolset R5.1 in www.st.com

[17] VEGA H.264 Buffer Analyzer (Version 6.8.1) from www.interrasystems.com

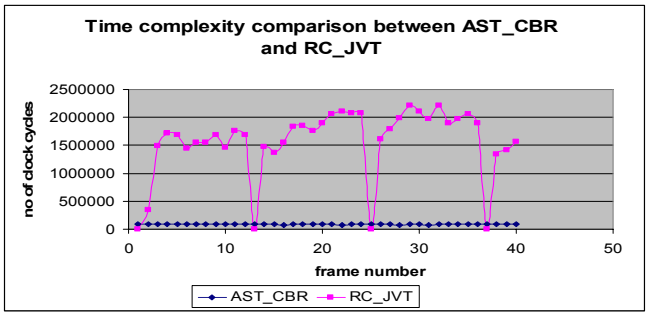

Figure 9 – Time complexity Comparison between AST-CBR and JVT, in the case of 40 frames films sequence, encoded at H264 baseline profile, 512 kbps with GOP length = 12.