

Title: Adaptive Reference Filtering for MVC with decoder complexity analysis

Status: Input Document to JVT

Purpose: Proposal

Author(s) or Contact(s): PoLin Lai, Antonio Ortega*
University of Southern California
Los Angeles, CA

Tel: +1 609 -987 7324

Email: Purvin.pandit@thomson.net
peng.yin@thomson.net

Purvin Pandit, Peng Yin, Cristina Gomila⁺
2 Independence Way,
Princeton, NJ – 08540, USA

Source: *University of Southern California
⁺Thomson

Abstract

JVT-W065 proposed an adaptive reference filtering approach (ARF) for inter-view prediction in multi-view video coding. It aimed at the problem of coding multi-view video that exhibits mismatches in frames from different views. This contribution is a follow-up of JVT-W065. It presents the detailed complexity analysis of ARF and compares it with Adaptive Interpolation Filter (AIF) and with the AVC sub-pel interpolation filters. Additionally, to reduce complexity while maintaining coding efficiency, it proposes to use simpler filters (3×3) with smaller support compared to JVT-W065 which used the 5×5 filters. The gains are asserted to range from 0.06 dB to 0.7 dB on anchor only coding (IPPPP). The asserted gain is larger for sequence with stronger focus mismatches.

1. Introduction

In multi-view video systems, scenes are captured simultaneously from multiple cameras and video frames from different views are prone to suffer from mismatches other than simple displacement. In JVT-W065 [1], we proposed an adaptive reference filtering approach (ARF) for inter-view prediction. It aimed at the problem of coding multi-view video that exhibits mismatches in frames from different views.

This contribution is a follow-up of our previous contribution. In this document, we present a detailed complexity analysis of the adaptive reference filtering (ARF) approach proposed in [1] and compare it with Adaptive Interpolation Filter (AIF) proposed in [2] and with the AVC sub-pel interpolation filters. Additionally, to reduce complexity while maintaining coding efficiency, we propose to use simpler filters (3×3) with smaller support compared to our previous proposal [1] which used the 5×5 filters.

2. Proposed Adaptive Filtering Approach For Inter-view Disparity Compensation

In our approach we use an initial disparity search to obtain the disparity field. This disparity field is used as an indication of depth and is used to classify current frame into different depth levels. We then associate a filter for each of the classified depth levels. The chosen filter is selected such that it minimizes the residual energy for all blocks within a given depth level class. The filter association process used in the experiments is based on the classification algorithms using the Gaussian Mixture Model (GMM) to separate blocks into classes (depth levels).

For details of the approach please refer to [1].

We now discuss how to select a filter for all blocks belonging to a given depth level class D_k . The equation used in the adaptive filter design can be written as:

$$\min_{\psi_k} \sum_{(x,y) \in D_k} \left(S_{x,y} - \sum_{j=-n}^n \sum_{i=-m}^m \psi_{i,j} R_{x+dv_x+i, y+dv_y+j} \right)^2$$

where S is the current frame to be encoded, R is the reference frame, the subscript denotes (x, y) the pixel position within a frame, $(x + dv_x, y + dv_y)$ is its corresponding disparity-displaced pixel in the reference frame. ψ_k is the filter corresponding to this depth-level class D_k .

The size and shape of 2D filters can be specified by changing m and n . We then apply adaptive filters directly to the reference frame to generate better matches. In our previous proposal [1], an odd-length 5×5 ($m = n = 2$) filter centered at the pixel to be filtered is employed. Symmetry constraints are imposed to reduce the number of unknowns in the adaptive filter estimation. Filters with more unknowns can be more efficient to compensate for residue energy. However, this comes at the expense of higher complexity as discussed in Section 3, and also having to estimate and transmit more filter coefficients. We have investigated other sizes with different types of constrain. In this document, we proposed to use a horizontal/vertical symmetric 3×3 filter ($m = n = 1$), in order to reduce the complexity while maintaining coding efficiency. In the following, we demonstrate these two types of filters along with the number of operations required to filter one pixel (denote as O_{ARF}).

Filter used in JVT-W065 (5×5 with hor/ver symmetry)

$$\psi = \begin{matrix} a_8 & a_6 & a_4 & a_6 & a_8 \\ a_7 & a_5 & a_3 & a_5 & a_7 \\ a_2 & a_1 & a_0 & a_1 & a_2 \\ a_7 & a_5 & a_3 & a_5 & a_7 \\ a_8 & a_6 & a_4 & a_6 & a_8 \end{matrix} \quad \text{Pixels: } \begin{matrix} A & B & C & D & E \\ F & G & H & I & J \\ K & L & M & N & O \\ P & Q & R & S & T \\ U & V & W & X & Y \end{matrix}$$

The number of operations required per pixel can be calculated as follows:

$$[M*a_0 + (L+N)*a_1 + (K+O)*a_2 + (H+R)*a_3 + (C+W)*a_4 + (G+I+Q+S)*a_5 + (B+D+V+X)*a_6 + (F+J+P+T)*a_7 + (A+E+U+Y)*a_8 + 2^{\text{bits}-1}] \gg \text{Bits}$$

$$O_{ARF} : 1+2+2+2+2+4+4+4+4 + 9 + 1 = 35 \text{ (9 additions in [1], and 1 shift)} \rightarrow \boxed{\text{Total 35 operations}}$$

The new proposed 3×3 with hor/ver symmetry:

$$\psi = \begin{matrix} a_3 & a_2 & a_3 \\ a_1 & a_0 & a_1 \\ a_3 & a_2 & a_3 \end{matrix} \quad \text{Pixels: } \begin{matrix} A & B & C \\ D & E & F \\ G & H & I \end{matrix}$$

The number of operations required per pixel can be calculated as follows:

$$[E*a_0 + (D+F)*a_1 + (B+H)*a_2 + (A+C+G+I)*a_3 + 2^{\text{bits}-1}] \gg \text{Bits}$$

$$O_{ARF} : 1+2+2+4 + 4 + 1 = \boxed{14 \text{ operations}}$$

3. Decoder Complexity Analysis

In this section, we aim to analyze the decoding complexity of ARF. ARF is applied to generate the integer pixel values of filtered references, which will be sequentially followed by H.264/AVC interpolation filters. Currently for the JMVM decoder, the subpel interpolation is performed *on-the-fly* for different blocks, i.e. upon receiving displacement vector for a particular block, decoder obtains the corresponding predictor integer pixels from the reference frame and then apply the interpolation. Thus, there are two possible approaches to implement the cascaded two-stage filtering of ARF and interpolation: The first approach is to generate ARF filtered references at frame level, and then apply on-the-fly interpolation at block level. The second approach is to apply both ARF and interpolation in an on-the-fly fashion. That is, for a given block, after decoding its `ref_idx` and displacement vector, corresponding pixels are obtained from the original (non-filtered) reference. The selected ARF filter (signaled by `ref_idx`) and interpolation will be performed on these integer pixels to generate the actual predictor. We will analyze the complexity of both approaches and recommend the on-the-fly approach, because it has less complexity. In [3], the complexity is analyzed in the unit block size of 4×4 . But since for MVC, most of the future application will be

for HD resolution, where the block size less than 8x8 is rarely used, we shall also analyze the complexity in the unit block size of 8x8.

3.1. Average number of operations with unit block size 4x4

As O_{ARF} denotes the operation per pixel for one ARF filter, and K is the number of ARF filters, the decoding complexity for the first approach (frame-wise ARF filtering and then on-the-fly H.264/AVC interpolations), in terms of average operation per pixel, will be $K \times O_{ARF} + \text{Average (H.264/AVC interpolation)}$. The analysis of the term ‘‘Average (H.264/AVC interpolation)’’ can be referred to [3]. We briefly summarize the result here (for the case of 4x4 unit block size):

	a	b	c
d	e	f	g
h	i	j	k
l	m	n	o

b (h) = Symmetric weighted average of 6 hor/ver integer pixels, 10 operations

a (c,d,l) = $(\text{Integer} + b + 1) \gg 1$, 10 operations for b + 3 = 13 operations

e (g,m,o) = $(b + h + 1) \gg 1$, 20 operations for b and h, + 3 = 23 operations

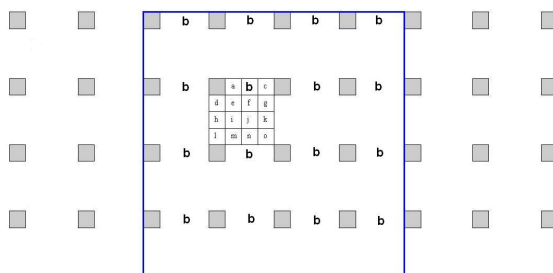
j = Symmetric weighted average of 6 vertical b values. Average 32.5 operations per pixel

f (i,k,n) = $(b + j + 1) \gg 1$, operations for j + 3. 35.5 operations per pixel.

As for the second approach (ARF also on-the-fly), at most one ARF filtering and one interpolation filtering need to be performed for a given block. Different subpel interpolations will require different number of integer pixels. Here we adopt the procedure in [3], in which the unit block size is 4x4, to analyze required ARF operations for each subpel position.

In the following, for a 4x4 block, the required pixels for a particular interpolation position are marked in gray. If the `ref_idx` indicates that ARF has to be used, each of the gray pixels has to be filtered by ARF with O_{ARF} before applying interpolation. Detailed analysis is provided below.

Interpolation position b (h)



Requires 36 integer pixels to obtain 16 b values

Operation per pixel: $(36 O_{ARF} + 16 \times 10) / 16 = 36 O_{ARF} / 16 + 10$

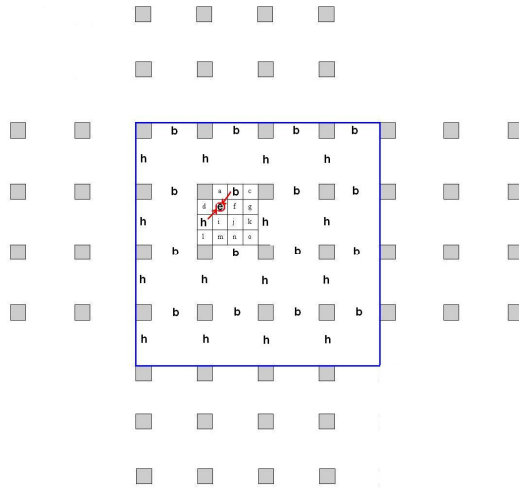
Interpolation position a (c, d, l)

$a = (\text{Integer} + b + 1) \gg 1$, exact same integer pixels are required as calculating b, so:

Operation per pixel: Operation of $(b) + 3 = 36 O_{ARF} / 16 + 13$

Interpolation position e (g, m, o)

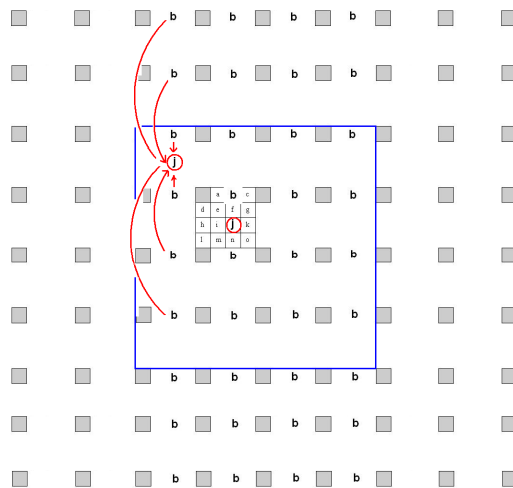
$e = (b+h+1) \gg 1$, so we need 16 b and h values, which requires 56 integer pixels as below:



Operation per pixel: $(56 O_{ARF} + 2 \times 16 \times 10) / 16 + 3 = 56 O_{ARF} / 16 + 23$

Interpolation position j

See the figure below for required 36 b values and corresponding 81 integer pixels.



Operation per pixel: $(81 O_{ARF} + 36 \times 10) / 16 + 10 = 81 O_{ARF} / 16 + 32.5$

Interpolation position f (i,k,n)

$f = (b + j + 1) \gg 1$. So after obtaining j, 3 more operations

Operation per pixel: $\text{Operation of } (j) + 3 = 81 O_{ARF} / 16 + 35.5$

Average operation per pixel (on-the-fly ARF plus H264 interpolation)

Assuming the displacement vectors have equal probabilities to be in any of the integer/subpel positions, the on average operations per pixel for a block using ARF filter would be:

$$[2 \times (b) + 4 \times (a) + 4 \times (e) + (j) + 4 \times (f) + O_{ARF}] / 16 = 861 O_{ARF} / (16 \times 16) + 21.16 = 3.36 O_{ARF} + 21.16$$

For a block use unfiltered reference, the complexity will be simply the H.264/AVC interpolations:

$$(2 \times 10 + 4 \times 13 + 4 \times 23 + 32.5 + 4 \times 35.5) / 16 = 21.16$$

Note that for an integer displacement vector, ARF still requires the corresponding pixels to be filtered, while there will be no interpolation operations for H.264/AVC. Thus we have added one more O_{ARF} term and consider the average over 16 possible positions (1 integer and 15 subpels).

Taking into account two possible cases of ARF references and non-filtered reference, the actual decoding complexity became:

$$P \times (3.36 O_{ARF} + 21.16) + (1-P) \times 21.16 \rightarrow 3.36 \times P \times O_{ARF} + 21.16$$

where P denotes the percentage of block selecting ARF filtered references. In our simulation, we have observed P ranging from 20% ~ 60%, across different bitrates and different sequences.

To compare the operation with H.264/AVC interpolation

$$R = \frac{3.36 \cdot P \cdot O_{ARF} + 21.16}{21.16}$$

By plugging-in the O_{ARF} numbers for 3x3 and 5x5 filters in Section 2 along with the P=20%~60%:

For hor/ver 5x5: R = 2.11 ~ 4.33

For hor/ver 3x3: R = 1.44 ~ 2.33 (ARF on-the-fly)

On the other hand, if we do frame-wise ARF followed by on-the-fly interpolations, and assuming 4 ARF filters are selected, the ratio will be:

$$R = \frac{K \cdot O_{ARF} + 21.16}{21.16} = \frac{4 \cdot O_{ARF} + 21.16}{21.16}$$

For hor/ver 5x5: R = 7.62

For hor/ver 3x3: R = 3.65

(Frame-wise ARF)

As a reference, the complexity analysis [3] for Adaptive Interpolation Filtering adopted in KTA [2] indicates a ratio about 3.34 when compared to H.264/AVC interpolation.

$$\frac{(2 \times 10 + 4 \times 13 + 4 \times 58 + 41 + 4 \times 55) / 16}{21.16} \approx \frac{35.3125}{21.16} \approx 1.67$$

To compare with H.264/AVC interpolation, there's one more aspect that should be noted. In order to hold the intermediate results of ARF before shifting, the operations introduced by O_{ARF} have to be in 32-bits length [3]. The H.264/AVC interpolation can be performed with 16-bits length. Thus, we included a weighting factor of 2 for the ARF operations. The ratio between the proposed approach and H.264/AVC interpolation become:

$$R = \frac{3.36 \cdot P \cdot O_{ARF} \cdot 2 + 21.16}{21.16}$$

By plugging-in the O_{ARF} numbers calculated for filters in Section 2 along with the above percentages, we get:

For hor/ver 5x5: R = 3.22 ~ 7.67

For hor/ver 3x3: R = 1.89 ~ 3.67 (ARF on-the-fly)

It shows that, by reducing the ARF filter size from 5x5 to 3x3, the complexity reduces into about half. On average for the filtering part (ARF plus interpolation), it is about 2~3 times as compared to H.264/AVC interpolation.

On the other hand, if we do frame-wise ARF followed by on-the-fly interpolations, and assuming 4 ARF filters are selected, the ratio will be:

$$R = \frac{K \cdot O_{ARF} \cdot 2 + 21.16}{21.16} = \frac{4 \cdot O_{ARF} \cdot 2 + 21.16}{21.16}$$

By plug-in the O_{ARF} numbers for 3x3 and 5x5 filters in Section 2:

For hor/ver 5x5: R = 14

For hor/ver 3x3: R = 6.29

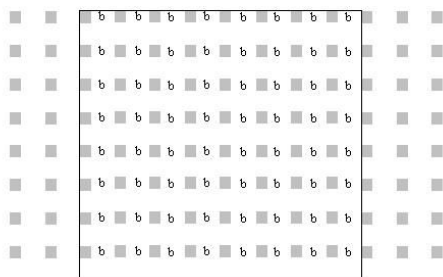
(Frame-wise ARF)

As a reference, the complexity analysis [3] for Adaptive Interpolation Filtering adopted in KTA [2] indicates a ratio about 3.34 when compared to H.264/AVC interpolation.

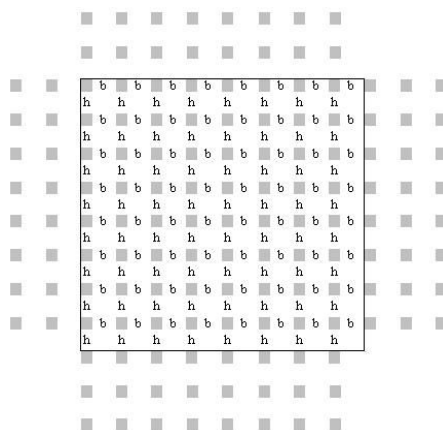
$$\frac{2 \times (2 \times 10 + 4 \times 13 + 4 \times 58 + 41 + 4 \times 55) / 16}{21.16} \approx \frac{70.625}{21.16} \approx 3.34$$

3.2. Average number of operations with unit block size 8x8

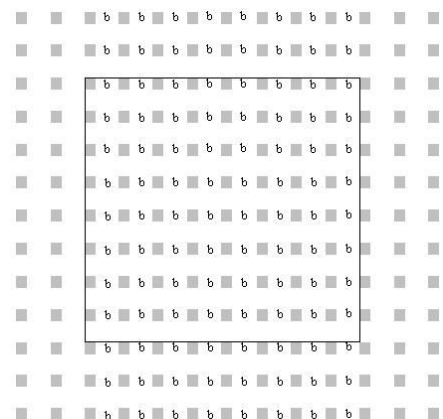
ARF introduces extra filter operations to the pixels required by interpolations. Thus, different unit block size will affect the analysis results. Following the procedure in Section 3.1, here we provide analysis for 8x8 block size:



Position b (h): $(104 O_{ARF} + 64 \times 10) / 64 = 104 O_{ARF} / 64 + 10$
Position a (c, d, l): $b + 3 = 104 O_{ARF} / 64 + 13$



Position e (g, m, o): $(144 O_{ARF} + 2 \times 64 \times 10) / 64 + 3 = 144 O_{ARF} / 64 + 23$



Position j: $(169 O_{ARF} + 104 \times 10) / 64 + 10 = 169 O_{ARF} / 64 + 26.25$
Position f (i,k,n): $j + 3 = 169 O_{ARF} / 64 + 29.25$

Average operation per pixel (on-the-fly ARF plus H264 interpolation)

$$[2 \times (b) + 4 \times (a) + 4 \times (e) + (j) + 4 \times (f) + O_{ARF}] / 16 = (2109 O_{ARF} / 64 + 307.25) / 16 = 2.06 O_{ARF} + 19.203$$

For a block use unfiltered reference, the complexity will be simply the H.264/AVC interpolations:

$$(2 \times 10 + 4 \times 13 + 4 \times 23 + 26.25 + 4 \times 29.25) / 16 = 19.203$$

Please note here that the average operation of H.264/AVC interpolation is decreased a little because different unit block size is used for analysis. As P denotes the percentage of block selecting ARF filtered references:

$$P \times (2.06 O_{ARF} + 19.203) + (1-P) \times 19.203 \rightarrow 2.06 \times P \times O_{ARF} + 19.203$$

To compare the operation with H.264/AVC interpolation

$$R = \frac{2.06 \cdot P \cdot O_{ARF} + 19.203}{19.203}$$

For 3x3 and 5x5 filters, with the P from 20% to 60%, we get:

$$\begin{aligned} \text{For hor/ver } 5 \times 5: R &= 1.75 \sim 3.25 \\ \text{For hor/ver } 3 \times 3: R &= 1.3 \sim 1.9 \quad (\text{ARF on-the-fly}) \end{aligned}$$

For the frame-wise ARF followed by on-the-fly interpolations, with 4 ARF filters (K=4), the ratio is

$$R = \frac{4 \cdot O_{ARF} + 19.203}{19.203}$$

Plug-in the O_{ARF} numbers calculated for filters in Section 2 we get:

$$\begin{aligned} \text{For hor/ver } 5 \times 5: R &= 8.29 \\ \text{For hor/ver } 3 \times 3: R &= 3.92 \quad (\text{Frame-wise ARF}) \end{aligned}$$

As a reference, for AIF the average number of operations per pixel is not affected by the unit size (8x8 or 4x4) being analyzed. However average H.264/AVC operation decreased to 19.203. Thus, the ratio of AIF compared to H.264/AVC interpolation will become [3]:

$$\frac{(2 \times 10 + 4 \times 13 + 4 \times 58 + 41 + 4 \times 55) / 16}{19.203} \approx \frac{35.3125}{19.203} \approx 1.84$$

When taken 32 bits into consideration, the ratio between the proposed approach and H.264/AVC interpolation become:

$$R = \frac{2.06 \cdot P \cdot O_{ARF} \cdot 2 + 19.203}{19.203}$$

By plug-in the O_{ARF} calculated for in Section 2, along with the P from 20% to 60%, we get:

$$\begin{aligned} \text{For hor/ver } 5 \times 5: R &= 2.5 \sim 5.51 \\ \text{For hor/ver } 3 \times 3: R &= 1.6 \sim 2.8 \quad (\text{ARF on-the-fly}) \end{aligned}$$

For the frame-wise ARF followed by on-the-fly interpolations, with 4 ARF filters (K=4), the ratio is

$$R = \frac{4 \cdot O_{ARF} \cdot 2 + 19.203}{19.203}$$

Plug-in the O_{ARF} numbers calculated for filters in Section 2 we get:

$$\begin{aligned} \text{For hor/ver } 5 \times 5: R &= 15.58 \\ \text{For hor/ver } 3 \times 3: R &= 6.83 \quad (\text{Frame-wise ARF}) \end{aligned}$$

As a reference, for AIF the average number of operations per pixel is not affected by the unit size (8x8 or 4x4) being analyzed. However average H.264/AVC operation decreased to 19.203. Thus, the ratio of AIF compared to H.264/AVC interpolation will become [3]:

$$\frac{2 \times (2 \times 10 + 4 \times 13 + 4 \times 58 + 41 + 4 \times 55) / 16}{19.203} \approx \frac{70.625}{19.203} \approx 3.68$$

3.3 Summary of Complexity Comparison

Table. 1 Operation Ratios compared to on-the-fly H.264/AVC interpolations, unit block size 4x4

	ARF on-the-fly (P = 20% ~ 60%)	Frame-wise ARF (K = 4)	AIF
Hor/ver 5x5 ARF	2.11 ~ 4.33	7.62	1.67
Hor/ver 3x3 ARF	1.44 ~ 2.33	3.65	

Table. 2 Operation Ratios compared to on-the-fly H.264/AVC interpolations, unit block size 8x8

	ARF on-the-fly (P = 20% ~ 60%)	Frame-wise ARF (K = 4)	AIF
Hor/ver 5x5 ARF	1.75 ~ 3.25	8.29	1.84
Hor/ver 3x3 ARF	1.3 ~ 1.9	3.92	

Table. 3 Ratios compared to on-the-fly H.264/AVC interpolations taken 32 bits into consideration, unit block size 4x4

	ARF on-the-fly (P = 20% ~ 60%)	Frame-wise ARF (K = 4)	AIF
Hor/ver 5x5 ARF	3.22 ~ 7.67	14	3.34
Hor/ver 3x3 ARF	1.89 ~ 3.67	6.29	

Table. 4 Ratios compared to on-the-fly H.264/AVC interpolations taken 32 bits into consideration, unit block size 8x8

	ARF on-the-fly (P = 20% ~ 60%)	Frame-wise ARF (K = 4)	AIF
Hor/ver 5x5 ARF	2.5 ~ 5.51	15.58	3.68
Hor/ver 3x3 ARF	1.6 ~ 2.8	6.83	

In Table 1 and 2, we list all the complexity comparisons of ARF with respect to H.264 subpel interpolation filter. For ARF which uses filter 3x3 and the on-the-fly approach, the additional operational complexity to H.264 interpolation is only 0.3 to 0.9. Since interpolation takes about 20% of decoding process, the total increase of complexity is 6% to 18%.

4. Simulation Results

In this simulation, we investigated the coding efficiency of ARF by simplifying the filter to 3x3 instead of 5x5 in our old proposal. We performed simulations using JMVM reference software tag JMVM_3_0 to encode multi-view video across views. We encoded the anchor frames only at GOP boundaries to evaluate different inter-view prediction schemes. Currently, ARF is implemented only for P pictures thus the anchor pictures use the I-P-P prediction structure.

The proposed adaptive reference filtering approach (ARF) is compared with JMVM_3_0 without illumination compensation. In our simulations, the maximum K (*number of filters*) allowed is 4. Thus, we also compare our method to JMVM with the number of reference frames set to 5.

The rate-distortion results are provided in Figure 1 to Figure 3. Simulations were performed with QP = 22, 27, 32, and 37 to obtain four rate points. For detailed results please see accompanying Excel file. The simulation results show that the proposed ARF coding scheme provides higher coding efficiency than current JMVM. Moreover, the performance of different methods varies significantly among different test sequences. Also the loss as a result of using a 3x3 filter compared to a 5x5 filter is not very significant and in most cases is limited to 0.1dB.

For the Ballroom sequence, the proposed ARF approach provides about a 0.25 ~0.3 dB gain over using 1, 3 and 5 references with a bitrate savings of about 5.7% ~ 7%. For Race1, which contains severe focus mismatches, the gain is higher, about 0.46 ~ 0.7dB with bitrate savings of about 9.4% ~ 13.7%. Exit and Uli show only a marginal improvement while Rena shows improvements of between 0.0dB ~ 0.37dB with a bitrate savings of 0.25% ~ 8.6%.

5. Conclusion

Based on the current experimental conditions, the results of the proposed ARF method are encouraging. A detailed complexity analysis of the tool at the decoder side was presented and a significant reduction in complexity was achieved compared to the previous proposal [1]. We recommend to JVT a core experiment be set up to further investigate the idea of adaptive reference filtering, especially for inter-view prediction.

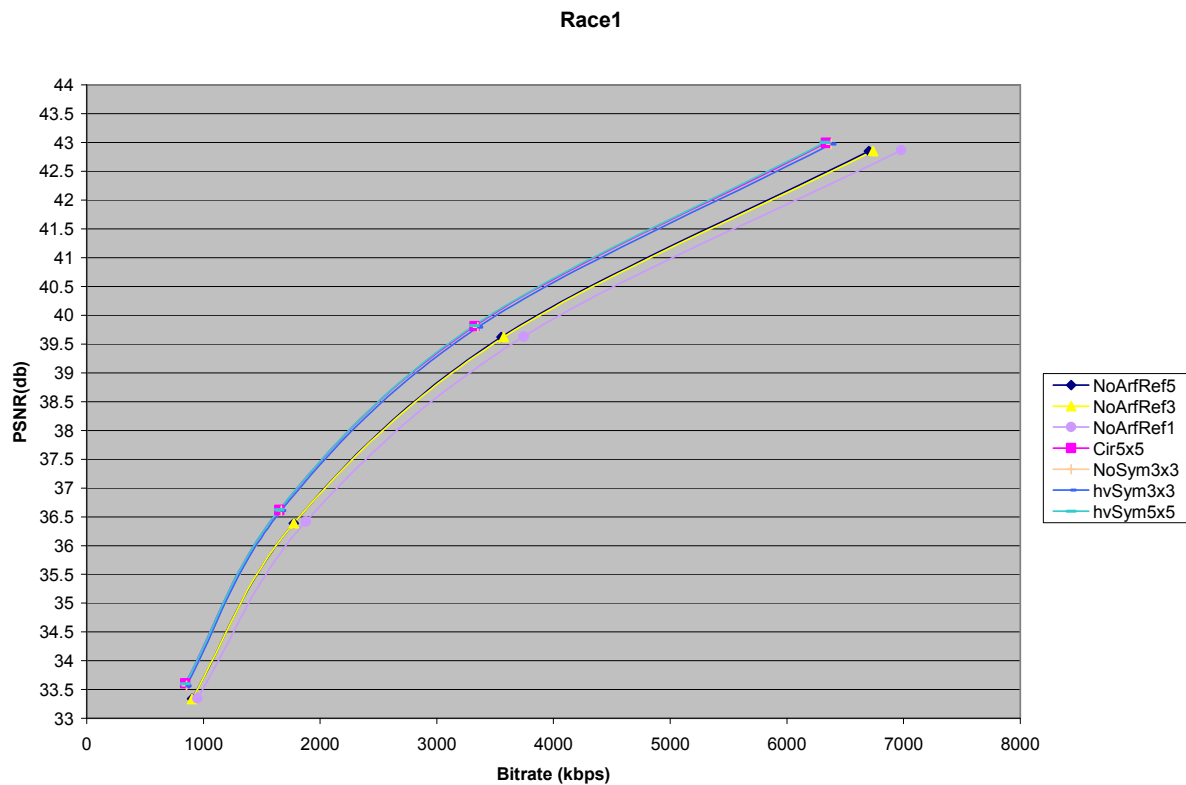


Figure 1: RD-plot for Race 1

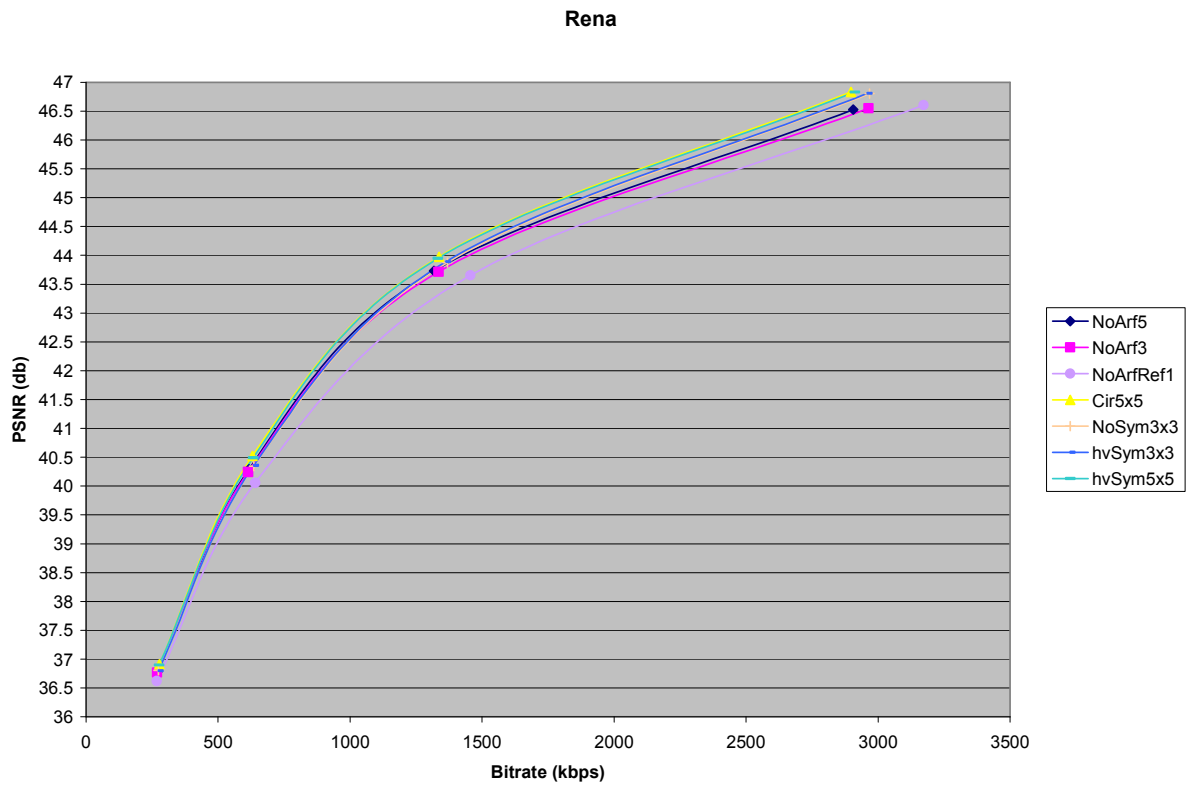


Figure 2: RD-plot for Rena



Figure 3: RD-plot for Ballroom

References

- [1] PL. Lai, A. Ortega, P. Pandit, P. Yin, C. Gomila, "Adaptive Reference Filtering for MVC", JVT-W065.doc, San Jose, USA, April 21-27, 2007.
- [2] Y. Vatis, B. Edler, D. T. Nguyen, J. Ostermann, "Two-dimensional non-separable Adaptive Wiener Interpolation Filter for H.264/AVC", ITU-T SGI 6/Q.6 Doc. MPEG05/VCEG-Z17, Busan, South Korea, April 2005.
- [3] Y. Vatis, J. Ostermann, "Comparison of Complexity between Two-dimensional non-separable Adaptive Interpolation Filter and Standard Wiener Filter ", ITU-T SGI 6/Q.6 Doc. MPEG05/VCEG-AA11, Nice, FR, 16-22 April, 2005.

(Append for Proposal Documents)

JVT Patent Disclosure Form

International Telecommunication Union
Telecommunication Standardization Sector



International Organization for Standardization



International Electrotechnical Commission



Joint Video Coding Experts Group - *Patent Disclosure Form*

(Typically one per contribution and one per Standard | Recommendation)

Please send to:

JVT Rapporteur Gary Sullivan, Microsoft Corp., One Microsoft Way, Bldg. 9, Redmond WA 98052-6399, USA
Email (preferred): Gary.Sullivan@itu.int Fax: +1 425 706 7329 (+1 425 70MSFAX)

This form provides the ITU-T | ISO/IEC Joint Video Coding Experts Group (JVT) with information about the patent status of techniques used in or proposed for incorporation in a Recommendation | Standard. JVT requires that all technical contributions be accompanied with this form. *Anyone* with knowledge of any patent affecting the use of JVT work, of their own or of any other entity (“third parties”), is strongly encouraged to submit this form as well.

This information will be maintained in a “living list” by JVT during the progress of their work, on a best effort basis. If a given technical proposal is not incorporated in a Recommendation | Standard, the relevant patent information will be removed from the “living list”. The intent is that the JVT experts should know in advance of any patent issues with particular proposals or techniques, so that these may be addressed well before final approval.

This is not a binding legal document; it is provided to JVT for information only, on a best effort, good faith basis. Please submit corrected or updated forms if your knowledge or situation changes.

This form is *not* a substitute for the *ITU ISO IEC Patent Statement and Licensing Declaration*, which should be submitted by Patent Holders to the ITU TSB Director and ISO Secretary General before final approval.

<u>Submitting Organization or Person:</u>	
Organization name	Thosmon Inc.
Mailing address	<u>2 Independence way</u> <u>Princeton, NJ 08540</u>
Country	USA
Contact person	Purvin Pandit
Telephone	+1 609-987-7324
Fax	+1 609-987-7299
Email	Purvin.pandit@thomson.net
Place and date of submission	Geneva, Switzerland, June 29 – July 6, 2007
<u>Relevant Recommendation Standard and, if applicable, Contribution:</u>	
Name (ex: “JVT”)	JVT-X
Title	Adaptive Reference Filtering for MVC
Contribution number	JVT-X060

(Form continues on next page)

Disclosure information – Submitting Organization/Person (choose one box)

2.0 The submitter is not aware of having any granted, pending, or planned patents associated with the technical content of the Recommendation | Standard or Contribution.

or,

The submitter (Patent Holder) has granted, pending, or planned patents associated with the technical content of the Recommendation | Standard or Contribution. In which case,

2.1 The Patent Holder is prepared to grant – on the basis of reciprocity for the above Recommendation | Standard – a free license to an unrestricted number of applicants on a worldwide, non-discriminatory basis to manufacture, use and/or sell implementations of the above Recommendation | Standard.

2.2 The Patent Holder is prepared to grant – on the basis of reciprocity for the above Recommendation | Standard – a license to an unrestricted number of applicants on a worldwide, non-discriminatory basis and on reasonable terms and conditions to manufacture, use and/ or sell implementations of the above Recommendation | Standard.

Such negotiations are left to the parties concerned and are performed outside the ITU | ISO/IEC.

2.2.1 The same as box 2.2 above, but in addition the Patent Holder is prepared to grant a “royalty-free” license to anyone on condition that all other patent holders do the same.

2.3 The Patent Holder is unwilling to grant licenses according to the provisions of either 2.1, 2.2, or 2.2.1 above. In this case, the following information must be provided as part of this declaration:

- patent registration/application number;
- an indication of which portions of the Recommendation | Standard are affected.
- a description of the patent claims covering the Recommendation | Standard;

*In the case of any box **other than 2.0** above, please provide the following:*

Patent number(s)/status _____

Inventor(s)/Assignee(s) _____

Relevance to JVT _____

Any other remarks: _____

(please provide attachments if more space is needed)

(form continues on next page)

Third party patent information – fill in based on your best knowledge of relevant patents granted, pending, or planned by other people or by organizations other than your own.

Disclosure information – Third Party Patents (choose one box)

3.1 The submitter is not aware of any granted, pending, or planned patents *held by third parties* associated with the technical content of the Recommendation | Standard or Contribution.

3.2 The submitter believes third parties may have granted, pending, or planned patents associated with the technical content of the Recommendation | Standard or Contribution.

For box 3.2, please provide as much information as is known (provide attachments if more space needed) - JVT will attempt to contact third parties to obtain more information:

3rd party name(s) _____

Mailing address _____

Country _____

Contact person _____

Telephone _____

Fax _____

Email _____

Patent number/status _____

Inventor/Assignee _____

Relevance to JVT _____

Any other comments or remarks: