ADAPTIVE FILTERING FOR VIDEO CODING WITH FOCUS CHANGE

PoLin Lai[a,b], Yeping Su, Peng Yin[b], Cristina Gomila[b] and Antonio Ortega[a]*

[a]Signal and Image Processing Institute, University of Southern California, Los Angeles, CA 90089
[b]Thomson Corporate Research, 2 Independence Way, Princeton, NJ 08540

ABSTRACT

We propose an adaptive filtering approach for coding video that exhibits localized camera focus changes, e.g., such that different portions of a video frame can undergo different blurriness / sharpness changes with respect to corresponding areas in frames used for prediction. First we obtain, for each macroblock, the filter that provides maximum reduction in residual energy, when applied to the reference macroblock before to motion compensation. Then, starting from the block-wise filter parameters, we divide the macroblocks into classes, by clustering macroblocks that have been assigned "similar" filters. Finally, for each class, a two-dimensional filter is designed to minimize the average residual energy for all macroblocks in the class. The resulting filters are applied to the reference frames to generate better matches for motion compensation. Simulation results shows that the proposed method provides up to 1 dB gain over current H.264 for certain sequences. As compared to H.264 with multiple reference frames, the coding gain is about 0.5 dB.

Index Terms— Adaptive filtering, focus change, motion compensation, video coding

1. INTRODUCTION

In most current video coding standards, hybrid coding techniques with block-based motion compensation and transform coding are employed. Motion compensation is applied to exploit temporal redundancy from frame to frame while the spatial redundancy within a frame is exploited by transform coding. When the objects in the video scene undergo only displacement with no other transformation, motion compensation will provides excellent coding efficiency, which can be further improved when variable block sizes and multiple reference frames are used, as in H.264 [1]. However, in situations where the pure displacement assumption is violated, the coding efficiency gains provided by motion compensation can be significantly lower.

In this paper, we consider situations where the video source contains camera focus changes. One often observed example of focus change occurs in dialog scenes, in which the camera shifts its focus from one character to another one at a different scene depth. The first character becomes blurred while the second gets sharpened. In other occasions, focus changes are created for transition during scene changes. To improve motion compensation performance in the presence of focus changes, Budagavi proposed blur compensation [2] for video coding, where a fixed set of blurring (lowpass) filters are used to generate blurred reference frames. This technique has two shortcomings for the scenarios we consider. First, the filter selection is made only at frame-level, i.e., applying different focus compensation filters to different parts of a frame is not possible. Here we specifically address adaptive local compensation. Second, this method relies on a predefined filter set, which does not include certain filters, e.g., sharpening filters (high frequency enhancement), which may be useful in some scenarios. Instead, our approach allows arbitrary filters to be generated for each frame, thus allowing blurring and sharpening filters to be used, with further extensions possible, e.g., using directional filters. Adaptive filtering methods have been proposed to address aliasing in generating subpixel references for motion compensation [3, 4]. After an initial motion search, macroblocks (MB) in the current frame are divided into groups based on the subpixel positions of their motion vectors. For each position, an adaptive filter is estimated. The subpixels of the reference frame will then be interpolated by these adaptive filters. In the final motion compensation, the encoder chooses the best match by testing different subpixels positions on the same reference frame [4]. This approach, which we will refer to as adaptive interpolation filtering (AIF), is specifically designed for subpixel interpolation filters and does not directly consider the localized focus mismatches that are the target of our work.

In this paper, we propose a new approach for video coding with focus changes, in which filters are adaptively estimated based on the difference between the reference frame and the current frame. Video frames are first divided into regions with different types of focus changes. For each region, a 2D filter is calculated to compensate the focus change by minimizing the residue energy. To provide better match, multiple versions of filtered reference are generated after applying the adaptive filters. For each block, encoder selects the filter that gives the lowest rate-distortion cost. The proposed method is described in Section 2. Simulation results based on H.264 are summarized in Section 3. Finally, conclusions are provided in Section 4.

2. ADAPTIVE FILTERING FOR FOCUS CHANGE COMPENSATION

Our goal is to enable locally adaptive compensation of focus changes. For each frame, we first perform motion search and estimate a simple focus change model for each MB. MBs with similar focus changes are grouped into classes; we call this process "filter association". For each class we then select a (more complex) filter that is optimized to minimize the residual energy for all macroblocks in the class. This filter is chosen to minimize:

$$\forall C_k: \quad \min_{\psi_k} \sum_{(x,y)\in C_k} \left(S_{x,y} - \psi_k * R_{x+mv_x, y+mv_y}\right)^2, \qquad (1)$$

where S is the current frame to be encoded, R is the reference frame, the subscript (x, y) is the pixel index within a frame, $(x + mv_x, y + mv_y)$ is its corresponding motion-displaced pixel in the reference frame, and C_k and ψ_k are, respectively, the set of pixels and the filter corresponding to class k ($*$ denotes the two-dimensional convolution.) The filters that satisfy (1) are Wiener filters that minimize

*The work was developed when the author was with Thomson Corporate Research, 2 Independence Way, Princeton, NJ 08540

the mean squared error (MMSE filters). Once these filters have been computed for each class, they are all applied to the reference frame. Then motion estimation and compensation is performed using both original and filtered frames as references, and each block is allowed to select a reference from any of the filtered frames. In the following subsections, we describe each step with more detail.

2.1. Filter association

The first step is to identify different types of focus changes in different parts of the current frame. An exhaustive approach could be to assign each block an adaptive filter that minimizes the matching error. This approach is optimal in the sense that for every block the residue energy is minimized. However, it will significantly increase the bitrate since we have to transmit filter coefficients for every single block. Instead, we consider procedures to classify blocks with similar focus changes. For instance, in the dialog example, blocks corresponding to the two characters (C_1 and C_2) will be associated with two different filters (ψ_1 and ψ_2), one for blurring and one for enhancement.

To achieve such classification, we considered two possible approaches. First, a set of predefined filters can chosen to operate on the reference frame. During an initial motion compensation, each block selects the filter that provides the lowest matching error. Blocks with similar filter selections can then be grouped into a class. This approach has a drawback in that it is possible that the predefined filter set is not complete enough to model all types of focus change within the frame. Thus, for blocks exhibiting focus changes that are not covered by the predefined filter set, we may fail to associate them with the right groups, leading to suboptimal compensation filters. Without prior knowledge of typical focus changes, or using a very large set of predefined filters, it will be hard to build a satisfactory filter set.

To avoid this problem, we investigate a second approach: During the initial motion compensation, a simple filter is estimated for each MB to minimize the residual energy (MMSE filter). The collection of all these MB-wise MMSE filters provides a more comprehensive description of various focus changes the current frame possesses. MBs are separated into groups based on the clustering of similar MB-wise filters. This procedure can be summarized as follows:

1. Initial motion search to obtain (mv_x, mv_y).
2. For each MB, calculate MMSE f_{mb} such that:

$$\min_{f_{mb}} \sum_{(x,y)\in MB} \left(S_{x,y} - f_{mb} * R_{x+mv_x,y+mv_y}\right)^2 \qquad (2)$$

   where the f_{mb} is in the form of $\begin{pmatrix} a & b & a \\ b & c & b \\ a & b & a \end{pmatrix}$

3. Classify f_{mb} into groups. Filter coefficients are considered as features for the classification algorithm. MBs are classified accordingly. For each class, one adaptive filter is associated with it to be estimated in the next stage.

The role of f_{mb} is to capture local focus changes from the reference frame to current frame. We selected a 3×3 filter with circular symmetry for two main reasons. First, MMSE estimation is more reliable when the number of equations is much greater than the number of unknowns. In our f_{mb}, there are only 3 variables with the number of equations equal to the number of pixels in the block. Second, larger filters with more variables will result in a much higher dimensional problem, which increases significantly the classification complexity. More importantly, this could also lead to an over-specified classification, which could be sensitive to filter variations and may not be suited to our goal of identifying rough class of focus changes within each frame.

Taking the coefficients as features, we can group the f_{mb} into classes. Such classification can be visualized by plotting each set of f_{mb} coefficients (a, b, c) as a point in 3D space. We have observed that the filter points all lie very closely to the $4a + 4b + c = 1$ plane (which we denote as P_f). This is reasonable since the MMSE system is attempting to find a weighted average for pixel values. By performing principal component analysis (PCA), we observe a system with a very insignificant third eigenvalue as compared to the first two (in the order of 10^{-15}), which indicates that the assumption that (a, b, c) belong to a plane is a reasonable one.

To select a classification tool, we performed the following study on f_{mb}: On plane P_f, we shift the filter coefficients away from the MMSE point (a, b, c) by $(\Delta a, \Delta b, \Delta c)$ and record how the MSE changes with different shifts. Statistics are gathered on a frame by frame basis. We observe that the increase in MSE away from the optimal point has different gradients in different directions. These findings suggest that the classification algorithm should take *directional information* into account, in addition to considering the distance between data points. Simply using the Euclidean distance to cluster the various filters into classes will not be appropriate as this would implicitly assume that the errors generated by changes in the filter coefficients are equal in all directions.

We propose to use classification algorithms based on multidimensional Gaussian Mixture Model (GMM) to separate f_{mb} into classes. GMM techniques incorporate covariance matrices such that the directional information can be modeled. In this paper, an unsupervised expectation-maximization (EM) clustering tool designed for GMM [5] is applied to classify f_{mb}. Based on the distribution of f_{mb} coefficients, it first constructs a GMM for the data points. The number of Gaussian components is estimated based on minimum description length (MDL) criteria [6] and the parameters (mean, covariance matrix, prior) of each Gaussian are estimated using an iterative EM algorithm [7]. Each Gaussian component is used to construct a multidimensional Gaussian probability density function (pdf) that models one class for classification. Likelihood functions can be calculated based on these Gaussian pdfs. Filters f_{mb} are classified into different groups by comparing their corresponding likelihood value in each Gaussian component.[1] An example of filter association results using the proposed method is provided in Fig.1.

Note that for illustration purposes, in this figure, the coefficients (a, b, c) of f_{mb} are projected onto the plane of P_f. Each feature component (a' and b') shown in the classification results is normalized. In this example excerpted from Time Machine movie trailer[2], camera focus is shifting from the back to the front. The face of the actor is getting blurred while his hand becoming more clear. The classification tool successfully separates these two groups, as can be observed in classes 1 and 2.

2.2. Class-adaptive filter selection

We now discuss how to select a filter for all blocks belonging to a given class. We rewrite (1) so that the optimization over all pixels in

[1] Refining processes can also be considered in the classification based on GMM, such as removing points with too low likelihood from the classes, or eliminating a class with too few points classified into it.

[2] "timemachine_320x160.mpg", http://timemachine.countingdown.com/

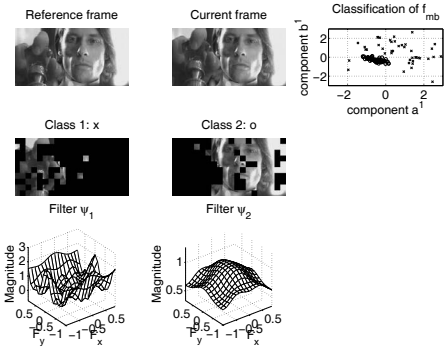

Fig. 1. Filter association results and the corresponding filters

class k is:

$$\min_{\psi} \sum_{(x,y)\in C_k} \left(S_{x,y} - \sum_{j=-n}^{n} \sum_{i=-m}^{m} \psi_{ij} R_{x+mv_x+i,y+mv_y+j} \right)^2 \tag{3}$$

Here we replace the convolution notation by explicitly expressing the filter operations. The size and shape of 2D filters can be specified by changing m and n. Constraints such as symmetry can be imposed to reduce the number of unknowns in adaptive filter estimation. Filters with more unknowns can be more efficient to compensate residue energy. However, this comes at the expense of having more filter coefficients to be transmitted. (For example, a circular symmetric 3×3 filter as in Section 2.1 contains 3 coefficients, while a full 3×3 matrix has 9 coefficients) In this paper, we demonstrate an *example* with 5×5 filters ($m = n = 2$) and the coefficients are constructed as:

$$\psi = \begin{pmatrix} a & b & c & b & a \\ d & e & f & e & d \\ g & h & j & h & g \\ d & e & f & e & d \\ a & b & c & b & a \end{pmatrix}$$

This can be viewed as a compromise between a full matrix and a circular symmetric one. For each group, a filter in the above form will be solved as specified by (3).

An example of the filters' frequency responses is also provided in Fig.1. For parts of the image that are getting enhanced such as around the hand (C_1), the filter ψ_1 emphasizes on higher frequency ranges so that the reference can be sharpened to create better match. For parts that are blurred (C_2), the corresponding filter ψ_2 is a low-pass filter with a Gaussian-shaped frequency response.

2.3. Motion compensation with local adaptive filtering

The obtained adaptive filters will be applied to the reference frame to generate better matches for motion compensation. In the reference picture list, the original unfiltered reference as well as multiple filtered references are stored. During the encoding process, each block in the current frame can select a block in any filtered or original reference frame, i.e., the one that provides the lowest rate-distortion cost, independently of whether the block was classified in a different class during the filter association process.[3]

To correctly decode the video sequence, filter selections of each block and the filter coefficients have to be transmitted to the decoder. Using the reference picture list as described above, the filter selection can easily be handled by signaling the reference frame index [1]. To encode the filter coefficients, in this paper, we directly extend the method proposed in [8, 9]. In Fig. 2 we provide the final filter selection result that corresponds to the example in Fig. 1.

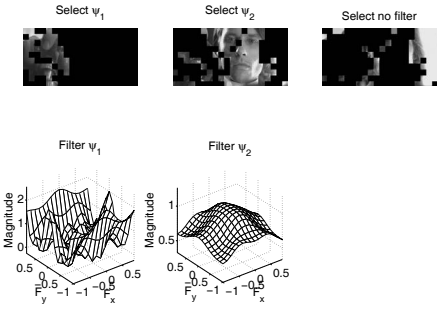

Fig. 2. Encoding selection with adaptive filtering

We can see in Fig. 2 that the hand and the face of the actor selected different filters to compensate the focus changes (ψ_1 and ψ_2). One interesting point to note is that for smooth regions such as the right most part of the current frame, the unfiltered reference is preferred. This is because for these plain regions, changing focus would not have much effect on the pixel values. We observed this same phenomenon at different frames as well.

3. SIMULATION RESULTS

We performed simulations based on H.264 video coding standard. The proposed approach is integrated with the JM 10.2 software [10] implementation of H.264. For solving the MMSE filters (both f_{mb} and ψ) and filtering the reference frame, we extended the code from [9]. The EM classification tool based on GMM [5] is combined with our program to take f_{mb} as input features. Reference frame management functions have been modified to store filtered reference frames. Finally, as described in Section 2.3, filter coefficients are encoded as in [8, 9].

The proposed approach is compared with the AIF and current H.264. Since we are keeping the original unfiltered reference, if the EM classification generates K classes for adaptive filters, there will be $N = K+1$ references in the reference list. In our experiments we have observed that 2 or 3 classes ($N = 3$ or 4) tend to be generated. Thus, we also compare our method to H.264 with the number of reference frame set to 5. The rate-distortion results of a sequence with strong localized focus changes[4] are provided in Fig. 3.

[3]Note that after this stage, the filter selection could be regarded as a new "filter association" C_k, and filters ψ_k could be estimated again based on MBs in different classes. Thus, the estimation of C_k and ψ_k can be carried iteratively until a stopping criterion is met. The complexity involved in such process will be fairly high. In this paper we limited ourselves to a single pass algorithm as described in Section 2.

[4]"fondue-multi.wmv", by Yi-Ren Ng, Light Field Photography project, Stanford Computer Graphics Lab. http://graphics.stanford.edu/papers/lfcamera/refocus/ (We encoded frame 126-170)

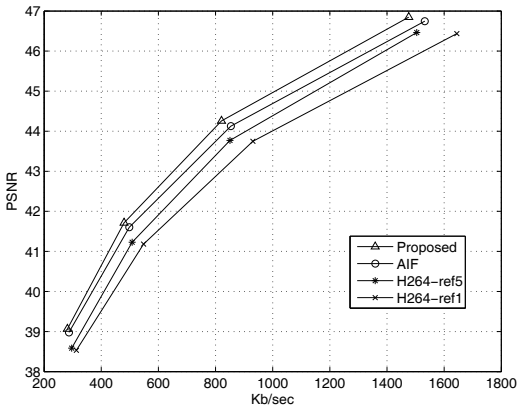

Fig. 3. Rate-Distortion comparison of different approaches

In this testing sequence, the camera focus is changing back and forth among people at different scene depths. It can be seen that the proposed adaptive filtering approach provides about 1 dB gain over H.264 with 1 reference, 0.5 dB gain over H.264 with 5 references, and around 0.2∼0.3 dB gain as compared to AIF. The results demonstrate that, when video undergoes local focus changes, adaptive filtering can be used to provide better reference for predictive coding. The proposed method and AIF both achieve higher coding efficiency than multiple reference method in H.264. Our approach is more effective in modeling the focus changes, with multiple versions of filtered references generated for motion compensation. In other video sequences excerpted from movie trailers, we also achieved 0.4∼0.7 dB gain over H.264 with 1 reference. The gain is much smaller when applying the proposed method to regular sequences with no focus changes. Thus, in a practical system, it would be desirable to develop tools to detect the existence of focus mismatch (e.g., applying appropriate criteria to residual blocks), so that mismatch compensation is only explored when a potential coding gain can be achieved.

We also analyzed the complexity of the proposed approach. Encoding time and motion estimation (ME) time were measured by a profiling tool in JM 10.2 software [10]. As compared to AIF, in which initial motion search and filter estimation are also involved, the total encoding time of our system is about 1.5 times as long. There are two factors of the increased complexity. One is the classification process used in filter association. The other is the ME loop over multiple filtered references. Instead, filters in AIF are imposed on the subpixel positions at a single reference frame. In Figure 4, for each method in Fig. 3, the ME time ratio as compared to H.264 with 1 reference is illustrated. The QP settings correspond to the rate-points in Fig. 3. While H.264 with 5 references has the highest ME computation cost, the ME time in our method is twice as long as in AIF. However, significant complexity reduction could be achieved by reusing motion information. References in our system are simply different filtered versions of the same frame. As we proceed from the original reference to filtered ones, a much refined search range based on previously computed motion could be applied. Predictive motion search exploiting such idea could help to bring down the ME time in our system close to AIF.

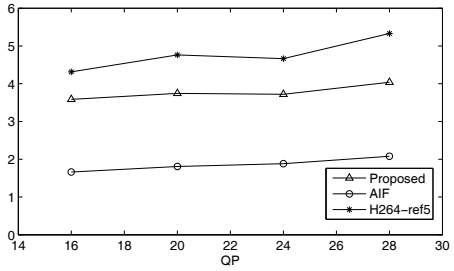

Fig. 4. ME time ratio as compared to H.264 with 1 reference

4. CONCLUSIONS

For video sequences containing camera focus changes, coding efficiency could be damaged as motion compensation fails to find good block correspondence. Furthermore, different regions within a frame may suffer different types of focus mismatches.

We proposed an encoding procedure based on adaptive filtering to compensate such discrepancy. It first captures the local variation of focus changes by estimating MB-wise filters. MBs with similar filters will be grouped together and associated with adaptive filters. EM classification algorithm with GMM basis is applied to consider directional variation of filter coefficients. This filter association approach is adaptive to the changes between the current frame and the reference frame.

Based on the filter association result, an adaptive filter is constructed for each group. Better matched references are generated by applying these adaptive filters. With the sequence we tested in our simulations, the proposed method provides higher coding efficiency as compared to the current H.264 with multiple reference frame and other adaptive filtering such as AIF.

5. REFERENCES

[1] T. Wiegand, G.J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. Circuits Systems and Video Technologies*, vol. 13 No.7, pp. 560–576, Jul. 2003.

[2] M. Budagavi, "Video compression using blur compensation," in *Proc. IEEE International Conference on Image Processing (ICIP)*, Genoa, Italy, Sep. 2005, pp. II.882–II.885.

[3] Thomas Wedi, "Adaptive interpolation filter for motion compensated prediction," in *Proc. IEEE ICIP*, Rochester, NY, Sep. 2002, pp. II.509–II.512.

[4] Y. Vatis, B. Edler, D. T. Nguyen, and J. Ostermann, "Motion-and aliasing-compensated prediction using a two-dimensional non-separable adaptive wiener interpolation filter," in *Proc. IEEE ICIP*, Genoa, Italy, Sep. 2005, pp. II.894–II.897.

[5] C. A. Bouman, "Cluster: An unsupervised algorithm for modeling Gaussian mixtures," *http://cobweb.ecn.purdue.edu/bouman/software/cluster/*, this version was released in Jul. 2005.

[6] J. Rissanen, "A universal prior for integers and estimation by Minimum Description Length," *Institute of Mathematical Statistics Journal: Annals of Statistics*, vol. 11, no.2, pp. 417–431, 1983.

[7] E. Redner and H. Walker, "Mixture densities, maximum likelihood and the EM algorithm," *SIAM Review*, vol. 26, no.2, Apr. 1984.

[8] Y. Vatis, B. Edler, I. Wassermann, D.T. Nguyen, and J. Ostermann, "Coding of coefficients of two-dimensional non-separable adaptive Wiener interpolation filter," in *Proc. SPIE Visual Communication and Image Processing*, San Jose, CA, Jul. 2005, vol. 5960, pp. 623–631.

[9] Y. Vatis, "Software implementation of adaptive interpolation filter," *http://iphome.hhi.de/suehring/tml/download/KTA/*, this software was released in Nov. 2005.

[10] Image Communication Group at Heinrich Hertz Institute Germany, "Software implementation of H.264: JM Version 10.2," *http://iphome.hhi.de/suehring/tml/index.htm*, this version was released in Jul. 2006.