Class-count reduction techniques for content adaptive filtering.

Citation for published version (APA):

DOI: 10.1109/ISCE.2009.5156908

Document status and date:
Published: 01/01/2009

Document Version:
Publisher’s PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:
• A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher’s website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the “Taverne” license above, please follow below link for the End User Agreement:
www.tue.nl/taverne

Take down policy
If you believe that this document breaches copyright please contact us at:
openaccess@tue.nl
providing details and we will investigate your claim.
Class-count Reduction Techniques for Content Adaptive Filtering

Hao Hu
Eindhoven University of Technology
Eindhoven, the Netherlands
Email: h.hu@tue.nl

Gerard de Haan
Philips Research Europe
Eindhoven, the Netherlands

Abstract—In the field of image/video enhancement, content adaptive filtering has shown superior performance over fixed linear filtering. The content adaptive filtering, first classifies the local image content based on different image features, such as structure and contrast. Then in every class, a least mean square (LMS) optimal filter is applied. A disadvantage of the concept is that many classes may be redundant, which leads to an inefficient implementation. In this paper, we propose and evaluate various class-count reduction techniques based on class-occurrence frequency, coefficient similarity and error advantage, which can greatly simplify the implementation without sacrificing much performance.

I. INTRODUCTION

Image/video enhancement often involves filtering. Content adaptive filtering has received considerable attention recently due to its superior performance over fixed filters [1]. A content adaptive filter first classifies local image content based on different image features, such as structure and contrast. Then in every class, a least mean square (LMS) optimal filter is employed. The structure adaptive filter was first proposed for image interpolation by Kondo [2], where only the image structure such as edge direction or luminance pattern is used for the classification. In applications to coding artifact reduction, additional features like block grid position [4], local contrast [5] and local variance [6], have been utilized. Incorporating more features in the classification improves the performance of the content adaptive filters, but also leads to an explosion of the class-count, many of which may be redundant.

For hardware implementation, a class-count reduction technique that allows a graceful degradation of the performance would be desirable. In this paper, we will investigate three options, which use class-occurrence frequency, coefficient similarity and error advantage, to reduce the number of classes. The results show that with the proposals the number of classes can be greatly reduced without serious performance loss.

The rest of the paper is organized as follows. Section 2 gives a brief introduction to the content adaptive filtering. In Section 3, three different class-count reduction techniques are presented. The evaluation of the techniques in the applications of image interpolation and coding artifact reduction are shown in Section 4. Finally, in Section 5, we draw our conclusions.

II. CONTENT ADAPTIVE FILTERING

In this section, we will introduce the framework of content adaptive filtering. The block diagram in Fig. 1 shows that the input pixel vector \(x\) from local image content within a filter aperture is first classified by image features such as local structure and contrast. A LMS-optimal linear filter is used to calculate the output pixel \(y\). The filter coefficients are obtained from an off-line supervised training using simulated input and reference output images. A typical structure classification is Adaptive Dynamic Ranging Coding (ADRC) [3]. The 1-bit ADRC code of every pixel is defined as:

\[
ADRC(x_i) = \begin{cases} 
0, & \text{if } x_i < \frac{x_{\text{max}} + x_{\text{min}}}{2} \\
1, & \text{otherwise} 
\end{cases}
\] (1)

where \(x_i\) is the value of pixels in the filter aperture and \(x_{\text{max}}, x_{\text{min}}\) are the maximum and minimum pixel value in the filter aperture. For applications like image interpolation, structure classification seems to be enough. For other application like coding artifacts reduction, the combination with other classification such as local contrast has been proven advantageous [5][6].

Fig. 1. The block diagram of the content adaptive filtering: the local image structure is classified using content classification and the filter coefficients are obtained from an offline training and stored in the LUT.

III. CLASS REDUCTION TECHNIQUES

Although the framework of content adaptive filtering leads to a simple hardware implementation, the size of the look-up-table may be significant. For example, the number of ADRC classes increases exponentially with the pixel count in the filter aperture. With a large number of classes, the method may not be efficient, i.e., there may be some redundancy...
in the classes. In this section, we explore three clustering techniques all capable to reduce the total number of classes, namely, class-occurrence frequency, coefficient similarity and error advantage. These techniques all use a similar scheme as follows. First one or more content classes will be clustered in a class-cluster. In every class-cluster an optimal linear filter is used. Every content class is assigned with a class-cluster label to indicate to which class-cluster it belongs. We use \( f(\cdot) \) to denote the labeling function that maps the content class \( k \) to the class-cluster number \( j \).

The filtering process is shown in Fig 2. First the local image content will be classified into content classes using different features, then the label look-up-table (LUT) is used to find out which cluster the content class belongs to. Then the corresponding filter is chosen for computing the output.

![Fig. 2. The block diagram of the class-reduced algorithm: the input vector is first pre-classified using content classification, then the content class is used to get the cluster number from the label LUT. Finally, the filter coefficients for the cluster is used for the filtering.](Image)

### A. Class-occurrence Frequency (CF)

One way to reduce the number of classes, is to merge the classes which are less important for the perceived image quality. The importance of a class is likely reflected to how often it occurs in an image. Therefore, we could count the occurrence frequency of every content class and hope that it can tell how important the content class is for the perceived image quality. Table I shows the percentage of most dominantly occurring classes in a set of sequences.

We would expect that if we merge the rarely occurring classes, the overall performance will suffer little. Suppose \( X_k \) denotes all the input vectors in content class \( k \). We sort the input vectors \( X_1, X_2, ..., X_k \) to \( X_{[1]}, X_{[2]}, ..., X_{[k]} \) from high to low by the occurrence frequency of its content classes. The \( M \) least frequent content classes will be merged into a cluster. The most popular classes, each will remain as a separate cluster. The labels will be:

\[
f(i) = \begin{cases} 
  [i], & \text{if } [i] < M \\
  M, & \text{otherwise}
\end{cases}
\]  

(2)

In this reduction technique, only one cluster includes more than one content classes. Therefore, in the hardware implementation, a number of comparators can be used instead of the more expensive label LUT. Fig. 3 shows a block diagram of using such comparators. \( M - 1 \) comparators contain the class codes \( C_{[1]}, C_{[2]}, ..., C_{[M-1]} \) which are sorted by its occurrence frequency. Once the input pixels are classified by different content classifications, the class code will be compared with the \( M - 1 \) most frequent occurring class codes. The comparison results are combined to a binary code to address the coefficient LUT.

![Fig. 3. The block diagram of using comparators for class-count reduction: \( M - 1 \) parallel comparators which contain the most dominantly occurring class codes.](Image)

### B. Coefficient Similarity (CS)

Another option to reduce the class-count is to examine the similarity between the filter coefficients obtained from the training in every content class. The filter coefficients directly show the filtering behavior and classes with similar coefficients can be merged. The similarity, here, is indicated by the the Euclidian distance between coefficient vectors. We propose to use the K-means algorithm to cluster the classes.

The clustering consists of the following steps:

1. Specify the number of clusters \( M \) according to the requirement and initialize the labels randomly.
2. Apply iterative steps to update the mean vector \( \mu^j_i \) in every cluster and the labels where \( i \) is the iteration number.
3. Repeat step 2 until the clustering converges. Convergence, here, means the labels do not change compared to the labels from the previous iteration.

#### 1. Specify the number of clusters \( \mu^j_i \) and clusters.

#### 2. Update the mean vector:

\[
\mu^j_i = \frac{\sum_{f(k) = j} W_k}{\sum_{f(k) = j} 1}
\]  

(3)

where \( W_k \) is the coefficient vector from the content class \( k \) and \( N_j \) is the number of content classes that belongs to the cluster \( j \).

#### 3. Update the labels with respect to the minimal distance from the mean vector:

\[
f(k) = \arg \min_j D(W_k, \mu_j).
\]  

(4)

where \( D(W_k, \mu_j) \) is the Euclidian distance between \( W_k \) and \( \mu_j \).
C. Error Advantage (EA)

The previous two approaches offer means to reduce the number of classes, however, they do not guarantee the minimization of total intra-cluster LMS estimation error. Therefore, a third technique is proposed to cluster the content classes with the respect to the error advantage of cluster LMS filters. Then the minimal total error can be achieved given a fixed number of clusters.

The clustering consists of the following steps:

1. Build the training set. As described in the Section 2, the vector pairs are collected from the simulated input and reference images, respectively. $S_k$ denotes the collection of all the vector pairs whose input vectors belong to content class $k$. Specify the number of clusters $M$ according to the requirement and initialize the labels randomly.

2. Apply the EM iterations [8] to the content classes to update the LMS filter coefficients $CW_j$ of all the clusters and cluster labels $f(\cdot)^i$, where $i$ is the number of iterations. The total intra-cluster mean square error (MSE) will decrease after every iteration until the clustering converges.

M-step: Obtain the cluster LMS filter coefficients $CW_j$ by LMS algorithm using the labels $f(\cdot)^{i-1}$.

$$CW_j^i = (\sum_{f(k)^{i-1}=j} S_{k,x}^x S_{k,x}^y)^{-1} \sum_{f(k)^{i-1}=j} S_{k,x}^x S_{k,y}^y$$

where $S_{k,x}$ and $S_{k,y}$ denote all input vectors and reference vectors from the content class $k$, respectively.

E-step: Evaluate the regression coefficients $CW_j$ on every content class and update the labels of sub-clusters with respect to minimal MSE.

$$f(k)^i = \arg \min_j E[(S_{k,y} - CW_j^{i-1}S_{k,x})^2]$$

3. Repeat step 2 until the clustering converges.

Comparing to the coefficient similarity approach, the iteration here involves much more calculations to evaluate all the cluster coefficients on the whole training set. Therefore, higher computation load and more training time is expected for the error advantage approach.

IV. RESULTS

In this section, we will evaluate the three class reduction techniques in the application to coding artifact reduction and image interpolation. In the experiment, we use a training set including about 2000 high resolution (1920 by 1080) images of various contents. For the evaluation, we use some test sequences shown in Fig. 4, which are not included in the training set. The test sequences are downgraded, as in the training, to generate the simulated input sequences. Then the input sequences are processed by different methods and the mean square error (MSE) between the original sequences and the processed ones are calculated and used for the subjective evaluation.

A. Combined coding artifact reduction and sharpness enhancement

In the application of combined coding artifact reduction and sharpness enhancement, it is concluded [5] that ADRC classification alone is not enough to distinguish between the coding artifacts and the real image structures. Therefore, one extra classification describing the contrast information in the filter aperture is added. Here we use the same filter setting as in [5]. The filter aperture is a diamond shape consisting of 13 pixels as shown in Fig. 1. An extra classification bit is used for the local contrast classification. The total number of classes is 8192.

In the experiment, the test sequences are first blurred, then compressed using JPEG compression, as in the training in [5], to generate the simulated input. Then the content adaptive filters with all the three class-count reduction techniques are evaluated using these sequences. For a fair comparison, we use the same number of clusters for the three class-count reduction techniques. An attractive number for hardware implementation, $M = 32$, is chosen for the experiment. For reference, we also include a fixed LMS filter which uses no classification.

Table II shows the MSE comparison of the evaluated methods. In terms of MSE score, one can see that three reduction techniques can reduce the number of classes by a factor of 256 with a modest increase of the MSE, compared to the MSE score of the fixed LMS filter. Among the three techniques, EA achieves the lowest MSE score, which it is expected, as it aims at minimizing the MSE.
TABLE II
MSE SCORES OF EVALUATED METHODS IN CODING ARTIFACT REDUCTION AND SHARPNESS ENHANCEMENT

<table>
<thead>
<tr>
<th>Sequence</th>
<th>No reduction</th>
<th>CF</th>
<th>CS</th>
<th>EA</th>
<th>Fixed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bicycle</td>
<td>76.9</td>
<td>88.0</td>
<td>87.0</td>
<td>85.1</td>
<td>97.7</td>
</tr>
<tr>
<td>Birds</td>
<td>15.7</td>
<td>17.0</td>
<td>16.5</td>
<td>16.9</td>
<td>19.8</td>
</tr>
<tr>
<td>Boat</td>
<td>87.5</td>
<td>91.1</td>
<td>90.7</td>
<td>90.2</td>
<td>96.5</td>
</tr>
<tr>
<td>Lena</td>
<td>40.4</td>
<td>43.0</td>
<td>42.6</td>
<td>42.3</td>
<td>46.0</td>
</tr>
<tr>
<td>Motor</td>
<td>141.6</td>
<td>150.2</td>
<td>149.1</td>
<td>148.0</td>
<td>157.5</td>
</tr>
<tr>
<td>Average</td>
<td>72.4</td>
<td>77.9</td>
<td>77.2</td>
<td>76.5</td>
<td>83.5</td>
</tr>
</tbody>
</table>

TABLE III
MSE SCORES OF EVALUATED METHODS IN IMAGE INTERPOLATION

<table>
<thead>
<tr>
<th>Sequence</th>
<th>No reduction</th>
<th>CF</th>
<th>CS</th>
<th>EA</th>
<th>Fixed</th>
<th>Atkins'</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bicycle</td>
<td>41.4</td>
<td>47.0</td>
<td>45.6</td>
<td>43.2</td>
<td>65.4</td>
<td>49.2</td>
</tr>
<tr>
<td>Birds</td>
<td>20.3</td>
<td>20.4</td>
<td>20.3</td>
<td>20.2</td>
<td>22.4</td>
<td>22.1</td>
</tr>
<tr>
<td>Boat</td>
<td>55.1</td>
<td>57.7</td>
<td>56.9</td>
<td>56.9</td>
<td>63.2</td>
<td>59.3</td>
</tr>
<tr>
<td>Lena</td>
<td>78.1</td>
<td>82.6</td>
<td>81.5</td>
<td>81.4</td>
<td>90.2</td>
<td>83.1</td>
</tr>
<tr>
<td>Motor</td>
<td>52.4</td>
<td>53.1</td>
<td>52.8</td>
<td>52.7</td>
<td>53.1</td>
<td>52.7</td>
</tr>
<tr>
<td>Average</td>
<td>49.4</td>
<td>52.2</td>
<td>51.4</td>
<td>51.2</td>
<td>58.9</td>
<td>53.3</td>
</tr>
</tbody>
</table>

Fig. 5 shows image fragments from the original sequence Bicycle, the simulated one, the processed ones by the original method without class reduction and with these reduction techniques. The three reduction techniques degrade the performance of the method without class reduction only little, while CS and EA show a better performance at suppressing the ringing artifacts than CF.

B. Image interpolation

For image interpolation, we apply the three class-count reduction techniques to Kondo’s method. As a comparison, we also choose Atkins’ method [7]. Atkins’ method is a content adaptive filtering method which applies soft probability-based classification and allows a flexible number of classes. For the evaluation, we use some test sequences shown in Fig. 4, which are not included in the training set. The test sequences first are down-scaled two times to generate the down-scaled version as the simulated input. Then Atkins’ method, the proposed class-count reduction methods and Kondo’s original method are evaluated using these sequences. For a fair comparison, we use the same number of clusters for the proposed method and Atkins’ method and the same aperture for the proposed methods and Kondo’s method. Since Atkins’ method performs best at the cluster number \( M = 100 \) [7], we use the same number here. Similar to the coding artifact reduction application, we also include a fixed filter for reference.

Table III shows the MSE comparison of the evaluated methods. In terms of MSE score, one can see that the proposed method outperforms Atkins’ method, while it is far less computationally expensive for both the classification and obtaining the filter coefficients. Compared to Kondo’s original method, the proposed methods only show a modest increase of the MSE score.

Fig. 6 shows image fragments from the original high resolution sequence Bicycle and processed ones by all the methods. All the three proposed methods render the lines in different directions correctly where Atkins’ method produces some staircase artifacts. Among them, EA and CS produce slightly smoother results at reconstructing the lines than the CF. They show more or less the same interpolation quality as Kondo’s original method, though the coefficient LUT size had been reduced nearly by a factor of 40 and only one label LUT with 1/26 the size of the original coefficient LUT and one extra fetch operation are needed. For CF, the comparators mentioned in Section III.A can be used instead of the expensive label LUT. The quality difference between CF and EA, CS is rather small. Therefore CF is more suitable for hardware implementation where the cost is more important.

V. Conclusion

In this paper, we have proposed three class-count reduction techniques, class-occurrence frequency, coefficient similarity and error advantage for the content adaptive filtering framework. In the applications of coding artifact reduction and image interpolation, it has been shown that these techniques can greatly reduce the number of content classes without sacrificing much performance and are promising for content adaptive filter with a large number of features. Among them, the coefficient similarity and error advantage approach produce the best result. Taking the cost into consideration, the class-occurrence frequency approach seems to be the best choice for implementation.

REFERENCES

Fig. 5. Image fragments the coding artifact reduction results on the Bicycle sequence: (A) Original, (B) Simulated input, (C) Without reduction, (D) Using occurrence frequency, (E) Using coefficient similarity, (F) Using error advantage, (G) Fixed filter.

Fig. 6. Image fragments from the image interpolation results on the Bicycle sequence: (A) Original, (B) Simulated input, (C) Without reduction - Kondo’s method, (D) Using occurrence frequency, (E) Using coefficient similarity, (F) Using error advantage, (G) Atkins’ method, (H) Fixed filter.