MATCHING-PURSUIT DICTIONARY PRUNING FOR MPEG-4 VIDEO OBJECT CODING

Yannick Morvan, Dirk Farin
University of Technology Eindhoven
5600 MB Eindhoven, The Netherlands
email: {y.morvan;d.s.farin}@tue.nl

Peter H. N. de With
LogicaCMG / Univ. of Technol. Eindhoven
5600 MB Eindhoven, The Netherlands
email: P.H.N.de.With@tue.nl

ABSTRACT
This paper describes how Matching Pursuit can be employed in an MPEG-4 video encoder as a replacement for the DCT at boundary-blocks to improve coding efficiency, while retaining backward compatibility. The key aspect of Matching Pursuit is that the video signal can be represented with less coefficients than with an orthogonal transform, due to the over-complete set of basis functions. However, Matching Pursuit cannot be integrated in a straightforward way into an existing encoder framework without loss of coding performance. Therefore, we modify the Matching Pursuit algorithm to increase the compression efficiency by a newly developed dictionary pruning scheme. Experimental results show that it can improve the obtained PSNR by up to 1 dB at object boundaries when compared to the basic Matching Pursuit.

KEY WORDS
Matching Pursuit, shape-adaptive transforms, MPEG-4 video coding, boundary blocks.

1 Introduction

Matching Pursuit (MP) [1] has been introduced as a technique to decompose a signal into a combination of basis functions. In contrast to orthogonal transforms like the DCT or wavelet transforms, MP uses an over-complete set of basis functions with the intention to represent an input signal with fewer coefficients. To emphasize the difference between the MP technique and transform coding, the basis functions are denoted as *atoms* and the set of atoms forming the set of basis functions is called a *dictionary*. There is a connection between the over-completeness of the dictionary and the sparsity of the signal representation. The larger the size of the dictionary, the fewer atoms are generally required to represent a signal. The MP technique decomposes a signal into a linear combination of atoms by employing an iterative approach. In each iteration, one coefficient is adapted to decrease the residual between the input signal and the reconstructed signal as much as possible.

Special applications of MP in existing encoder architectures enable increased coding efficiency while retaining backward compatibility. However, Matching Pursuit can usually not be integrated in a straightforward way into an existing encoder framework without loss of coding performance. For this reason, we have modified the Matching Pursuit algorithm to increase the compression efficiency by using a newly developed dictionary pruning scheme.

In this paper, we consider the example application of boundary block coding as it is employed in the object-oriented MPEG-4 video coding standard. The MPEG-4 video coding algorithm is based on the DCT and applied to independent blocks of 8×8 pixels each. However, video objects with an arbitrary shape also comprise boundary blocks in which some of the pixels are transparent. Since these transparent pixels are masked out in the decoder, their value can be chosen arbitrarily. The MPEG-4 standard recommends to fill the undefined pixels with copies of the pixel values at the object border. This pixel replication, called *padding*, usually results in different textures for the object region and the padded region. Unfortunately, this can decrease the compression efficiency, because coding a mixture of textures is less efficient than coding a single texture. This disadvantage can be circumvented with the following two approaches.

Special region-based transforms have been proposed that operate on an arbitrary-shaped region instead of a square block. One proposed transform is the *Shape Adaptive DCT* (SA-DCT) [2], which combines the foreground pixels by shifting them into a connected region. This arbitrarily-shaped region is then transformed using a row-by-row sequence of variable length 1-D transforms. The advantage of this approach is that only as many coefficients are generated as there are foreground pixels in the block. Unfortunately, the pixel shifting usually decreases the correlation between neighboring pixels, resulting in a higher bit-rate.

The second approach exploits the idea that the transparent pixels can be set to arbitrary values. To obtain a minimum-cost representation of the foreground texture, it is clear that only foreground pixels should be taken into account. The value of the background pixels are derived in such a way that the foreground region is optimally compressed. To consider only foreground pixels in the DCT coefficients computation, we mask the DCT basis functions by setting the basis functions positions that correspond to transparent pixels, equal to zero (see Figure 1). By applying this mask, the set of DCT basis functions loses its orthonormality so that we get a set of non-orthonormal basis functions [3].

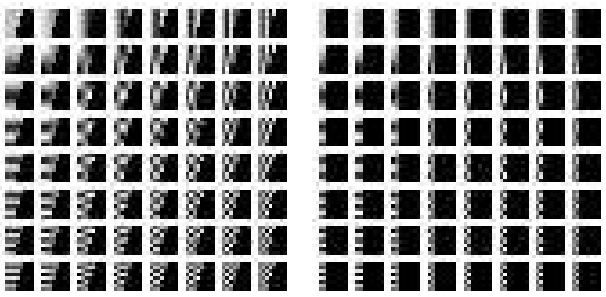
The loss of orthonormality implies that the coefficients cannot be obtained with a simple transform. As an alternative [3, 4], the Matching Pursuit algorithm will give suitable coefficients. With an appropriate scaling, the computed coefficients can be used without modification at the decoder side, thereby offering backward compatibility. Similar to the SA-DCT, we base our computations on the foreground texture only, while still using the 8×8-DCT transform. On the other hand, we have found that this may give an efficiency loss, since the MP algorithm may generate coefficients with a different probability distribution than the DCT, leading to inefficient entropy coding.

Therefore, we propose to adapt the coding scheme so that the MP technique can be integrated with an increased efficiency. This advantage is obtained by modifying the dictionary used by the MP algorithm such that coefficients are generated in a distribution comparable to the original DCT transformation. It will be shown that our proposal gives up to 1 dB PSNR improvement.

The paper is structured as follows. Section 2 will briefly introduce the Matching Pursuit algorithm. In Section 3, we will express the redundancy of a dictionary by describing its *coherence* parameter. Section 4 describes a new general dictionary pruning technique to adapt the properties to the required MPEG-4 statistics for coding. Results are presented in Section 5 and the paper concludes in Section 6.

2 Matching Pursuit technique

MP decomposes a signal into a linear combination of basis functions (atoms), selected from an over-complete dictionary. The selection of atoms and the corresponding computation of a coefficient is carried out with a greedy algorithm. It selects in each iteration one atom for modification, in order to minimize the residual signal in the atom-related dimension. We will briefly introduce the MP algorithm using the following notation.



(a) (b)

Figure 1. Two example dictionaries for arbitrary-shaped object coding. The atoms are based on the DCT basis functions, where the transparent pixels have been set to zero.

- I is the input data vector, i.e. the image block to code.
- $\mathcal{D} = \{d_0, d_1, \ ... \ , \ d_{M-1}\}$ is the dictionary of atoms. A single atom d_k is selected from the set $\mathcal{D}$ by $\mathcal{D}(k) = d_k$.
- The set $\{\alpha_0, \ ... \ , \ \alpha_{M-1}\}$ contains the coefficients corresponding to the atoms in $\mathcal{D}$.
- In accordance with existing literature [1], we denote $R^n I$ as the residual signal at iteration n, where we start with $R^0 I = I$.

The algorithm operates on an input signal I and a predefined dictionary $\mathcal{D}$. At the n-th iteration (starting with $n = 0$), the atom $\mathcal{D}(\gamma_n)$ is selected, so that $\mathcal{D}(\gamma_n)$ yields the largest reduction of the current residual $R^n I$. The largest reduction can be obtained with the atom that is most collinear to the residual $R^n I$. Thus, we select at iteration n the atom with index γ_n according to

$$\gamma_n = \underset{k=0,\ldots,M-1}{\arg\max} \left(\frac{|< \mathcal{D}(k), R^n I >|}{\| \mathcal{D}(k) \|} \right). \tag{1}$$

Note that the same atoms can be selected multiple times, since the greedy algorithm can only make an optimal decision for a single iteration. After this, the coefficient corresponding to the selected atoms is computed with

$$\hat{\alpha} = \frac{< \mathcal{D}(\gamma_n), R^n I >}{\| \mathcal{D}(\gamma_n) \|^2}, \tag{2}$$

and this value is used to update $\alpha_{(\gamma_n)}$ by

$$\alpha_{(\gamma_n)} \leftarrow \alpha_{(\gamma_n)} + \hat{\alpha}. \tag{3}$$

Finally, we obtain the new residual by subtracting the contribution of the selected atom with index γ_n from the current residual $R^n I$:

$$R^{n+1} I \leftarrow R^n I - \hat{\alpha} \mathcal{D}(\gamma_n). \tag{4}$$

This process is repeated until the residual is smaller than a pre-defined threshold. As a result after N iterations, we obtain an alternative representation of the signal I as

$$I = \sum_{k=0}^{M-1} \alpha_k \mathcal{D}(k) + R^N I. \tag{5}$$

Note that the sum serves as an approximation of I, whereas $R^N I$ is the remaining residual.

3 Expressing redundancy of the dictionary

For each boundary block, the dictionary is filled with basis functions that are masked with the shape information of the current block. Since the mask varies for different boundary

blocks, the dictionary also varies between blocks. Consequently, also the amount of redundancy in the dictionary varies from block to block. Let us try to capture the redundancy of a dictionary. In [5], the concept of a so-called *coherence* parameter was introduced. In the sequel, we reuse this parameter, albeit with a new definition given by

$$\mu = \max_{j \neq k} | < \mathcal{D}(j), \mathcal{D}(k) > | \ . \tag{6}$$

This definition of the *coherence* parameter indicates how similar the atoms in a dictionary are. The concept of similarity was also explored in [6]. In our case, the coherence is higher for small boundary blocks with only a few opaque pixels. Consequently, we have a lot of background pixels, leading to a more over-complete dictionary and thus more similarity between atoms. In the extreme case where a block only contains one opaque pixel, all the atoms are identical up to a scaling factor. However, this means that any atom can equally represent the input pixel. For less trivial cases as shown in Figure 1(b), it is possible to see that the dictionary can include very similar atoms. This redundancy is a clear disadvantage, because it results in a higher bit-cost to code the selection between these redundant atoms. It would be preferable to keep only a smaller set of atoms which has the same signal composition capability, but which enables us to be more efficient in coding the atom selection. Obviously, we would like to keep atoms giving the lowest bit-cost.

Since we are considering to use MP in an existing encoder framework, the cost to code a specific atom is predefined. In the case of DCT coefficient coding, high-frequency coefficients are more expensive to code than low-frequency coefficients, because coefficients are ordered according to increasing frequency and the position is coded with a run-length code. Larger run-lengths require larger Huffman code word-lengths. Consequently, we propose to obtain a subset from the atoms in the original dictionary by removing redundant atoms. The derivation of redundant atoms is discussed in detail in the next section. From the derived set of redundant atoms, we remove the atom with the highest frequency index, since this atom is assumed to require the highest number of code bits.

4 Dictionary subset selection

This section describes two algorithms for reducing the redundancy of a given dictionary. In [7], a scheme has been proposed which selects independent atoms based on a modified Gram-Schmidt orthogonalization procedure with pivoting technique. We adopt a similar idea, but we develop it in a different way as follows.

An atom can be removed from the dictionary if it is very similar to other atoms, or if it can be replaced by a combination of a few other atoms. When a group of redundant atoms is found, we can remove atoms from the dictionary, while still being able to represent all possible input vectors with the remaining atoms. Since the number of bits required to code each individual atom is variable, we remove the atoms that have a higher bit-cost. After all redundant atoms have been removed, we obtain a reduced dictionary $\mathcal{D}' \subseteq \mathcal{D}$ that has signal composition capabilities which are close to the original dictionary with a smaller number of atoms. Moreover, the coding cost for the computed coefficients is smaller because costly atoms were replaced with atoms from $\mathcal{D}'$ that have lower bit-cost. The trade-off for finding the optimal cost is discussed in the next section.

We present two algorithms for removing redundant atoms where the difference between both algorithms is the condition that decides when an atom is considered to be redundant. In the first algorithm (*sub-space redundancy removal*), an atom is considered redundant if it can be replaced with a linear combination of other atoms. The result of this algorithm is a non-redundant dictionary with as many atoms as pixels in the shape mask. While the advantage is that redundant atoms with high coding-cost are removed, we lose the benefits of an over-complete dictionary. For this reason, we propose a second algorithm (*vector redundancy removal*), which preserves over-completeness in the dictionary by only removing an atom if it is almost equal to another atom.

4.1 Sub-space redundancy

The first algorithm selects a sub-dictionary $\mathcal{D}'$ from the original dictionary $\mathcal{D}$ such that the sub-dictionary consists of linearly independent atoms. This is carried out by removing all atoms from $\mathcal{D}$ that are a linear combination of other atoms. However, in practice, linear dependence can not be detected easily, because small arithmetic inaccuracies can make a set of dependent atoms independent. To decide if an atom $\mathcal{D}(k)$ is independent from a set of atoms $\mathcal{D}(p)$ with $p = 0, ..., n$, we try to represent $\mathcal{D}(k)$ as a linear combination of atoms from $\{\mathcal{D}(p)\}$ and a remaining error vector δ^k, where δ^k is as small as possible [8]. To this end, we determine weighting factors β_p^k, such that

$$\mathcal{D}(k) = \sum_{p=0}^{n} \beta_p^k \mathcal{D}(p) + \delta^k, \tag{7}$$

with the constraint that δ^k is orthogonal to each $\mathcal{D}(p)$. This means that

$$< \delta^k, \mathcal{D}(p) >= 0 \qquad \text{for} \qquad p = 0, ..., n. \tag{8}$$

To compute the weighting factors β_p^k, we project Equation (7) onto the atoms $\mathcal{D}(p)$, which results in the linear system

$$\begin{array}{l} \mathcal{C}_{0k} = \beta_0^k \mathcal{C}_{00} + \beta_1^k \mathcal{C}_{01} + ... + \beta_n^k \mathcal{C}_{0n} \\ \mathcal{C}_{1k} = \beta_0^k \mathcal{C}_{10} + \beta_1^k \mathcal{C}_{11} + ... + \beta_n^k \mathcal{C}_{1n} \\ ... \\ \mathcal{C}_{nk} = \beta_0^k \mathcal{C}_{n0} + \beta_1^k \mathcal{C}_{n1} + ... + \beta_n^k \mathcal{C}_{nn} \end{array} , \tag{9}$$

where

$$\mathcal{C}_{ij} = < \mathcal{D}(i), \mathcal{D}(j) > . \tag{10}$$

Note that we assume that the atoms $\mathcal{D}(p)$ are normalized. To account for the numerical inaccuracies, we consider $\mathcal{D}(k)$ linearly dependent if $||\delta^k||$ does not exceed a tolerance ξ. This is equivalent to

$$\delta^k = ||\mathcal{D}(k) - \sum_{p=0}^{n} \beta_p^k \mathcal{D}(p)|| \leq \xi. \quad (11)$$

Based on the technique to check linear dependency of one atom to a set of other atoms, we will now define an algorithm that constructs a sub-dictionary $\mathcal{D}' \subseteq \mathcal{D}$ that only contains independent atoms. Moreover, because the coding-cost of atoms is not equal, the indices of the selected atoms should be as small as possible. This is achieved by iterating through the atoms of dictionary $\mathcal{D}$ in the order of increasing cost and adding the atom to $\mathcal{D}'$, if it is independent to the previous atoms (see Figure 2).

Input:

- Redundant dictionary $\mathcal{D}$.
- Tolerance $\xi \approx 0.0001$.

Output: Non-redundant sub-dictionary $\mathcal{D}'$.
Initialization: $\mathcal{D}' \leftarrow \{\mathcal{D}(0)\}$.
for *k=1 to M-1* **do**
  Compute β_p^k using Eq. (9).
  if $\| \mathcal{D}(k) - \sum_{p=0}^{k-1} \beta_p^k \mathcal{D}'(p) \| > \xi$ **then**
    $\mathcal{D}' \leftarrow \mathcal{D}' \cup \{\mathcal{D}(k)\}$
  end
end

Figure 2: Sub-space redundancy removal algorithm.

Since coefficients with lower cost are checked prior to coefficients with high cost, they are earlier added to $\mathcal{D}'$. Consequently, $\mathcal{D}'$ and $\mathcal{D}$ have the same ability to reconstruct a given signal. However, as over-completeness has been removed, more coefficients are required to represent a signal using $\mathcal{D}'$. It is difficult to find the optimal trade-off between single atom versus a linear combination of other atoms, because optimality depends on the data and the mapping onto code bits. The optimal algorithm is still under study. Reducing over-completeness can lead to slower convergence, since there is less freedom in the convergence path. To combine the advantages of a modified dictionary and an over-complete dictionary, we require an algorithm that removes costly atoms only if the atoms are clearly redundant. In the following section, we propose an algorithm having this property.

4.2 Vector redundancy

The vector redundancy algorithm maps single atoms into single vectors. The aim is to remove costly atoms from the original dictionary $\mathcal{D}$ that are similar to an already accepted single atom with a lower cost. Again, the reduced dictionary is called $\mathcal{D}'$. The amount of atoms that are removed can be adjusted using a similarity threshold. In general, this does not decrease the dictionary size so much that its over-completeness would be removed. Like in the previous algorithm, we verify the similarity in the order of increasing coding-cost. This leads to a reduced dictionary $\mathcal{D}'$ that preferably contains atoms with low coding-cost.

We define the similarity metric between two atoms as the absolute angle $|\theta|$ between their basis functions. Since with normalized vectors it holds that

$$|\cos(\theta)| = | < \mathcal{D}(i), \mathcal{D}(j) > |, \quad (12)$$

we can use the inner product as a measure of similarity. Using this definition, two collinear basis functions will have a unity similarity value. This value reduces to zero for orthogonal basis functions. Since we want to exclude almost equal atoms only, we choose a similarity threshold $\zeta \approx 0.9$. The vector redundancy algorithm is summarized in Figure 3.

Input:

- Dictionary $\mathcal{D}$
- Threshold $\zeta \approx 0.9$

Output: Non-redundant sub-dictionary $\mathcal{D}'$
Initialization: $\mathcal{D}' \leftarrow \{\mathcal{D}(0)\}$.
for *k=1 to M-1* **do**
  if $\max_{n=0,\ldots,k-1} | < \mathcal{D}(k), \mathcal{D}(n) > | < \zeta$ **then**
    $\mathcal{D}' \leftarrow \mathcal{D}' \cup \{\mathcal{D}(k)\}$
  end
end

Figure 3: Vector redundancy removal algorithm.

Let us examine the example in Figure 4. We have five atoms $\mathcal{D}(1), \ldots, \mathcal{D}(5)$, where the index refers to the coding-cost for each atom. It can be seen that some atoms are very similar, so that one of the similar atoms can be removed. Evidently, we remove the atom with the higher coding-cost. In this example, we exclude atom $\mathcal{D}(3)$ and $\mathcal{D}(5)$.

5 Experimental results

We have evaluated the performance of both algorithms. Experiments were carried out using the first 100 frames of the *Foreman*, *Kettle*, and *Stefan* sequences in CIF resolution. Since we apply our new algorithms to boundary blocks only, we compare results on the basis of those blocks. We used an MPEG-4 encoder framework, where only intra-frame coding of the texture was applied. The presented bit-rates involve the texture data only; the shape information is not included as it is equal for both algorithms. During preliminary experiments, we have found that the vector redundancy algorithm clearly outperforms the sub-space redundancy algorithm. Therefore, in the sequel, we focus on

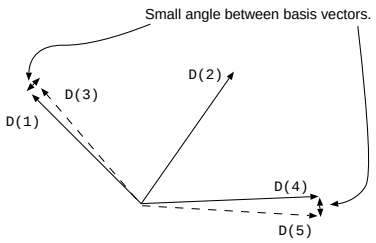
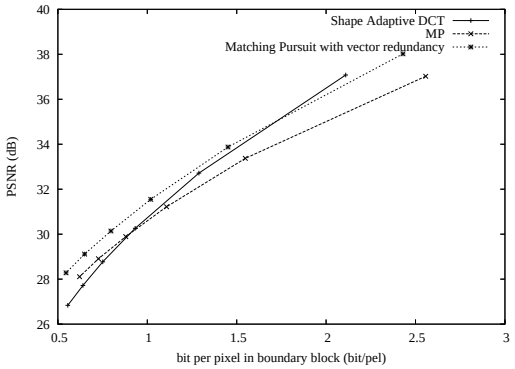


Figure 4. Geometrical interpretation of vector redundancy removal. Dashed vectors are removed because they are quasi-collinear to a lower cost vector (the frequency order k is appended to each vector).

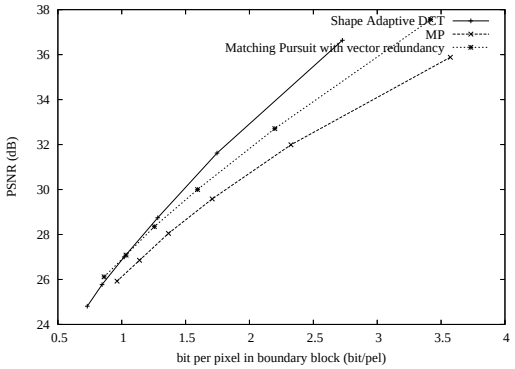
the vector redundancy algorithm only. We implemented the original Matching Pursuit (MP) algorithm and an MP algorithm with the vector redundancy algorithm. The threshold ζ of the vector redundancy algorithm was set to $\zeta = 0.9$. For comparison, we have also included the results of the SA-DCT algorithm for the same conditions. The results are shown in Figure 5, which portrays PSNR curves as a function of the bit rate.

Let us first compare the MP algorithms with the SA-DCT technique. An advantage of the SA-DCT algorithm is that all generated coefficients are placed in the top-left corner of the coefficient matrix. This results in short run-lengths of zeros to code the coefficient positions. However, the disadvantage is that the correlation between pixel values is reduced (see the discussion in Section 1). The video sequence determines which of these aspects has the largest influence. In the *Foreman* and *Stefan* sequences, the boundary blocks present limited texture details only, whereas the *Kettle* sequence includes boundary blocks with sharp texture edges due to an imprecise segmentation mask. Consequently, for normal video sequences (*Foreman* and *Stefan*), the vector redundancy algorithm yields better results than the standard SA-DCT for low bit-rate. Because of the sharp edges in the *Kettle* sequence, many high frequencies occur, so that the coding-cost of coefficients is higher than with SA-DCT (SA-DCT transfers coefficients to the low-frequency area). Figure 5 also shows that the vector redundancy algorithm clearly outperforms regular MP in all cases. In the *Foreman* and *Stefan* sequence, 1 dB at 1 bit per pixel is gained, and 0.5 dB in the *Kettle* sequence. Figure 6, depicting the first frame of the *Foreman* sequence, shows a gain on texture details in boundary blocks coded with the vector redundancy algorithm (Fig. 6(c)) compared to regular MP (Fig. 6(d)). The vector redundancy removal algorithm shows a better performance than regular MP, because it preserves the over-completeness of the dictionary and only removes the atoms that are clearly redundant.

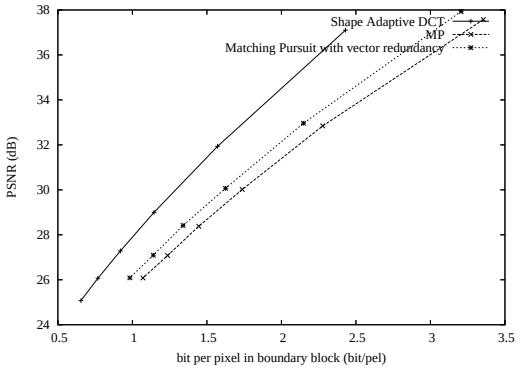
We also conducted experiments with various thresholds ζ. We have found out that the results do not depend much on the value of ζ as long as its value is close to unity. The reason for this is that we obtain the largest improve-



(a)



(b)

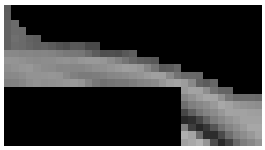


(c)

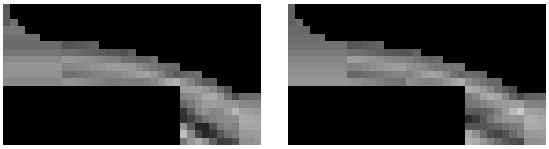
Figure 5. Rate-distortion curve for object boundary blocks Intra-frame coding in *Foreman* 5(a), *Stefan* 5(b) and *Kettle* 5(c) sequences. Measurements achieved on the luminance plane (Y) with MPEG-4 VLC table.

ments from the removal of very similar atoms. Setting ζ to a smaller value would increase the risk of removing required atoms so that a good signal composition cannot be guaranteed anymore.

(a)



(b)



(c) (d)

Figure 6. (a) Video object plane from the first frame of the *Foreman* sequence. (b) Magnified view of the marked area. (c) Coded boundary blocks with vector redundancy algorithm. (d) Regular MP. Both results are obtained at 0.6 bit/pixel.

6 Conclusion

We have presented an algorithm for reducing the dictionary redundancy of MP that extends Matching Pursuit coding for further compression of boundary blocks in object-oriented MPEG-4 processing. This algorithm prunes the MP dictionary such that not only the number of atoms is reduced, but it also adapts the probability distribution of the generated coefficients to the statistics that were assumed in the existing DCT-based framework. The vector redundancy algorithm improves regular MP with 0.5 to 1 dB for various scenes. The experiments showed that the compression efficiency improvement occurs because the algorithm removes clearly redundant atoms from the MP dictionary, while it does not reduce the over-completeness of the dictionary. The algorithm uses a similarity metric and a corresponding threshold ζ, which can be applied for quality control. Usually, for normal objects, the proposed algorithm is more efficient than SA-DCT at low bit rates (up to 1 bit/pixel). An extra value of our algorithm is that it can be integrated into an already existing DCT-based encoder system so that it is backwards compatible.

References

[1] S. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Transactions on Signal Processing*, vol. 41, no. 12, pp. 3397–3415, 1993.

[2] T. Sikora and B. Makai, "Shape-adaptive dct for generic coding of video," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 5, pp. 59–62, 1995.

[3] Y. Shishikui and S. Sakaida, "Region support dct (rs-dct) for coding of arbitrarily shaped texture," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, pp. 320–330, May 2002.

[4] A. Kaup and T. Aach, "A new approach towards description of arbitrarily shaped image segments," in *International Workshop on Intelligent Signal Processing and Communication Systems*, pp. 543–553, 1992.

[5] D. Donoho and X. Huo, "Uncertainty principles and ideal atomic decomposition," *IEEE Transactions on Information Theory*, vol. 47, pp. 2845–2862, 2001.

[6] J. A. Tropp, "Greed is good: Algorithmic results for sparse approximation," ICES Report 03-04, The University of Texas at Austin, February 2003.

[7] L. Rebollo-Neira, "Dictionary redundancy elimination," *IEE Proceedings - Vision, Image and Signal Processing*, vol. 151, pp. 31–34, 2004.

[8] R. R. Pati Y.C. and P. Krishnaprasad, "Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition," vol. 1 of *Conference Record of The Twenty-Seventh Asilomar Conference on Signals, Systems, and Computers*, pp. 40–44, 1993.