

INSERTION OF IRREGULAR-SHAPED LOGOS IN THE COMPRESSED DCT DOMAIN

Nuno Roma¹ and Leonel Sousa²

¹Nuno.Roma@inesc-id.pt

²las@inesc-id.pt

Instituto Superior Técnico / INESC-ID
Department of Electrical Engineering
Rua Alves Redol, No. 9 - 1000-029 Lisboa - PORTUGAL
Tel. +351 21 3100300 - Fax: +351 21 3145843

Abstract: This paper addresses the problem of the insertion of irregular-shaped objects, such as logos or subtitles, in video signals in the compressed DCT domain. To achieve this objective, a different approach from the usual pixel domain compositing operation is adopted, in order to avoid the presence of undesired semi-transparent regions around the inserted objects. The transposition of this technique to the frequency domain is presented, as well as the inclusion of a logo insertion module in the architecture of a compressed domain video transcoder.

1. INTRODUCTION

Recently, one has assisted to the proliferation of advanced video services and multimedia applications, where video coding algorithms, such as MPEG-x or H.26x, have been proposed to broadcast video information in digital form. However, once video signals have been compressed, one frequently faces the need for further manipulations on such compressed bit-streams. Consequently, *video transcoding* has recently emerged as a new research area concerning a set of manipulation techniques such as space scaling, frame skipping, bit-rate adjustment, etc.

Despite this wide range of video manipulations, the insertion of objects, such as logos and open subtitles (henceforward simply designated by “*logos*”) in the compressed domain has faced a growing interest by broadcasting television networks to provide the possibility of inserting their own logos and subtitles on pre-encoded video streams [1].

Furthermore, recent developments on video transcoders have shown that significant advantages concerning the computational complexity of the algorithms can be achieved by fully operating in the frequency domain [2]. In fact, not only does it avoid the implementation of both the forward Discrete Cosine Transform (DCT) and its inverse (IDCT), but it also takes advantage of the presence of a large number of zeros in the quantized blocks to heavily reduce the data manipulation rate [3].

Up until now, most insertion algorithms were based on the compositing operation proposed by Porter [4]. However, despite its simplicity, when irregular-shaped objects (such as letters) are inserted, it gives rise to an undesired semi-transparent rectangular region around the logo, corresponding to the area that is actually processed by the insertion algorithm. In this paper, a different insertion technique is proposed, by restricting the Porter’s algorithm [4] to the logo area. However, the application of this technique in the compressed domain will require the usage of the multiplication-convolution relationship for the DCT, since the simple linear combination of Porter’s algorithm will not be applicable any more.

2. LOGO INSERTION IN THE PIXEL DOMAIN

Logo insertion in the pixel domain can be performed by combining the pixels of the background image $b(n_1, n_2)$ with the logo $\ell(n_1, n_2)$ to obtain the output image $p(n_1, n_2)$. This operation is usually expressed as a linear combination of the form [4]:

$$p(n_1, n_2) = \alpha \cdot \ell(n_1, n_2) + (1 - \alpha) \cdot b(n_1, n_2), \quad (1)$$

where the α factor determines the transparency of the logo. In particular, when $\alpha = 1$, all pixels of the background image are replaced by the logo, giving rise to an opaque overlapping of the logo over the input image.

The main disadvantage of this method is emphasized when Irregular-Shaped Logos (ISL) are inserted. In fact, since most video coding algorithms perform their processing on blocks with $N \times N$ pixels (usually $N = 8$), the insertion of logos whose shape and position do not coincide with the defined block geometry and grid implies the extension of the original logo area to an integer multiple of $N \times N$ blocks. This extension is usually performed by defining a transparency color (T), whose value is assigned to all pixels of this set of blocks that do not belong to the original ISL. However, three different regions in the output image usually arise from this extension:

- the pixels corresponding to the original logo, where $p_1(n_1, n_2) = \alpha \cdot \ell(n_1, n_2) + (1 - \alpha) \cdot b(n_1, n_2)$;
- the transparent pixels of the extended blocks that do not belong to the original logo, where $p_2(n_1, n_2) = \alpha \cdot T + (1 - \alpha) \cdot b(n_1, n_2)$;
- the blocks that do not contain any pixel of the original logo, where $p_3(n_1, n_2) = b(n_1, n_2)$.

Independently of the considered value for T , this scheme gives rise to an undesired semi-transparent rectangular region around the logo, where $p(n_1, n_2) = p_2(n_1, n_2) \neq b(n_1, n_2)$. An example of this phenomenon is illustrated in fig. 1(a), where eq. 1 has been applied to insert the logo using $\alpha = 0.5$.



(a) Porter's technique [4]. (b) Proposed technique.

Figure 1. Pixel domain insertion of an ISL ($\alpha = 0.5$).

This undesired semi-transparent region can be avoided if eq. 1 is restricted to the pixels of the original logo. However, this will imply the usage of segmentation techniques to isolate the original logo from the rest of the pixels of the block.

Nevertheless, the formalism previously described for the linear combination can still be applied, if the concept of transparency mask is introduced and defined as:

$$m(n_1, n_2) = \begin{cases} 1 & , (n_1, n_2) \in \text{ISL} \\ 0 & , (n_1, n_2) \notin \text{ISL} \end{cases} \quad (2)$$

Hence, eq. 1 becomes:

$$\begin{aligned} p(n_1, n_2) &= [\alpha \cdot m(n_1, n_2)] \odot \ell(n_1, n_2) \\ &\quad + [1 - \alpha \cdot m(n_1, n_2)] \odot b(n_1, n_2) \quad (3) \\ &= \phi(n_1, n_2) + \psi(n_1, n_2) \odot b(n_1, n_2) \end{aligned}$$

where \odot denotes pixel-wise multiplication. In this equation, $\phi(n_1, n_2)$ and $\psi(n_1, n_2)$ represent constant matrixes that are solely dependent on the considered logo. Hence, they can be pre-computed and stored in memory. In fig. 1(b) it is illustrated the result of this technique using $\alpha = 0.5$. As it can be seen, the undesired semi-transparent region is not present around the ISL any more.

3. LOGO INSERTION IN THE DCT DOMAIN

The pixel domain insertion algorithm presented in eq. 1 can be directly applied in the compressed domain by using the orthogonality properties of the DCT. Since α and $(1 - \alpha)$ are scalars and the DCT is linear in relation to the scalar-product and addition operations, one obtain:

$$P(k_1, k_2) = \alpha \cdot L(k_1, k_2) + (1 - \alpha) \cdot B(k_1, k_2), \quad (4)$$

where $X(k_1, k_2) = DCT[x(n_1, n_2)]$.

However, since most video coding algorithms perform the DCT on $N \times N$ blocks, this relation cannot be used if the undesired semi-transparent region around the ISL is intended to be avoided. In fact, since $[\alpha \cdot m(n_1, n_2)]$ and $[1 - \alpha \cdot m(n_1, n_2)]$ of eq. 3 are matrixes, the pixel-wise multiplication will have to be replaced by a convolution in the DCT domain.

3.1. Multiplication-Convolution Theorem in the DCT Domain

Although the DCT is closely related to the Discrete Fourier Transform (DFT), the multiplication-convolution theorem for the DCT has been established much after the corresponding relationship for the DFT.

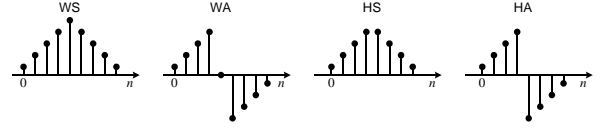


Figure 2. Types of symmetry of an infinite sequence.

In fact, despite the several attempts to establish this relation [5], a complete and more consistent formalization was only presented relatively recently [6, 7]. In particular, Martucci [6] presented a formalized and detailed description of the symmetric convolution operation for the entire family of discrete sine and cosine transforms. He considered the DCT and the DST as special cases of the so called Generalized Discrete Fourier Transform (GDFT), which operate on infinite sequences that are strictly periodic or antiperiodic, with period N . These infinite sequences are also classified according to the types of symmetry that they present, as it is illustrated in fig. 2. Symmetry within any sequence must be of one of the following four types:

WS - Whole-sample symmetry,

WA - Whole-sample anti-symmetry,

HS - Half-sample symmetry,

HA - Half-sample anti-symmetry,

where the designations *whole-sample* and *half-sample* refer to the position of the point of symmetry: either coincident with one of the samples or at a theoretical half-way between two samples.

A given finite sequence $x(n)$ can be converted into an infinite sequence by symmetrically extending each of its ends using one of the above four possible ways and continuing that extension indefinitely. Such symmetric extension is usually denominated by concatenating the mnemonics of the symmetry types used at each of its ends (e.g.: WSWS, HAHA, WAHS, etc.).

Hence, the particular case of the DCT-II transform is characterized by taking a length- N fraction of a HSHS periodic sequence as input to obtain a length- N fraction of a WSWA anti-periodic sequence as output. According to Martucci [6], the symmetric convolution in the DCT-II domain between two length- N WSWA sequences X and Y is defined as follows:

$$Z(k) = X(k) \otimes Y(k) = W_N \left(\tilde{X} \textcircled{S} \tilde{Y} \right), \quad (5)$$

where \tilde{X} and \tilde{Y} are symmetric length- $2N$ WSWA extended sequences of X and Y , defined as:

$$\tilde{X}(k) = \begin{cases} 0 & , k = 0 \\ \hat{X}(N - k) & , k = 1 \dots (N - 1) \\ \hat{X}(k - N) & , k = N \dots (2N - 1) \end{cases} \quad (6)$$

where $\hat{X}(k) = \frac{X(k)}{C(k)}$, with $X(k) = DCT[x(n)]$, and

$$C(k) = \begin{cases} 1/\sqrt{2} & , k = 0 \\ 1 & , k = 1 \dots (N - 1) \end{cases} \quad (7)$$

The symbol \textcircled{S} denotes the skew-circular convolution, defined as shown in eq. 8 (in the following page), where:

$$S(k) = \begin{cases} 1 & , k \in [0, (2N - 1)] \\ -1 & , \text{otherwise} \end{cases} \quad (9)$$

$$\begin{aligned}
\tilde{X}(k) \circledast \tilde{Y}(k) &= \frac{1}{\sqrt{2N}} \cdot C(k) \cdot \left[\sum_{m=0}^k \tilde{X}(m) \tilde{Y}(k-m) - \sum_{m=k+1}^{2N-1} \tilde{X}(m) \tilde{Y}(k-m+2N) \right] \\
&= \frac{1}{\sqrt{2N}} \cdot C(k) \cdot \left[\sum_{m=0}^{2N-1} \tilde{X}(m) \tilde{Y}[\text{mod}_{2N}(k-m)] \cdot S(k-m) \right]
\end{aligned} \tag{8}$$

$$\begin{aligned}
\Psi(k_1, k_2) \circledast B(k_1, k_2) &= \\
&= \frac{1}{2N} C(k_1) C(k_2) \left[\sum_{m_1=0}^{2N-1} \sum_{m_2=0}^{2N-1} \tilde{\Psi}(m_1, m_2) \tilde{B}[\text{mod}_{2N}(k_1-m_1), \text{mod}_{2N}(k_2-m_2)] \cdot S(k_1-m_1) \cdot S(k_2-m_2) \right]
\end{aligned} \tag{11}$$

$$\tilde{X}(k_1, k_2) = \begin{cases} 0 & , k_1 = 0 \text{ or } k_2 = 0 \\ \hat{X}(N-k_1, N-k_2) & , k_1 = 1 \dots (N-1), k_2 = 1 \dots (N-1) \\ \hat{X}(k_1-N, N-k_2) & , k_1 = N \dots (2N-1), k_2 = 1 \dots (N-1) \\ \hat{X}(N-k_1, k_2-N) & , k_1 = 1 \dots (N-1), k_2 = N \dots (2N-1) \\ \hat{X}(k_1-N, k_2-N) & , k_1, k_2 = N \dots (2N-1) \end{cases} \tag{12}$$

and $W_N(k)$ is a length- N rectangular window to extract the representative samples out of the base period of the result of the convolution.

3.2. Logo Insertion

Having defined the multiplication-convolution theorem for the DCT, the application of eq. 3 in the compressed domain is stated as follows:

$$P(k_1, k_2) = \Phi(k_1, k_2) + \Psi(k_1, k_2) \circledast B(k_1, k_2) \tag{10}$$

where $P(k_1, k_2) = \text{DCT}[p(n_1, n_2)]$, $\Phi(k_1, k_2) = \text{DCT}[\phi(n_1, n_2)]$ and $\Psi(k_1, k_2) = \text{DCT}[\psi(n_1, n_2)]$. The above 2D convolution is computed as shown in eq. 11, where $C(k)$ and $S(k)$ were defined in eq. 7 and eq. 9, respectively, and $\tilde{X}(k_1, k_2)$ is a $2N \times 2N$ symmetric WSWA extended sequence defined as shown in eq. 12.

4. APPLICATIONS TO TRANSCODING

As it was previously referred, the main target application for logo insertion in the compressed DCT domain is video transcoding. One of the most simple and well-known architectures of a compressed domain video transcoder is presented in fig. 3. The logo insertion module is placed between the decoder part and the encoder part of the transcoder, where the data corresponding to the decoded image is available in the compressed domain. Consequently, the DCT motion compensated video data is first convolved with $\Psi(k_1, k_2)$, corresponding to the transparency mask, and added with the data corresponding to the ISL $\Phi(k_1, k_2)$ (see eq. 10). As it was previously mentioned, both $\Psi(k_1, k_2)$ and $\Phi(k_1, k_2)$ depend only on the considered logo and can be pre-computed and stored in memory (see fig. 3).

However, an efficient implementation of this architecture is only possible if some important issues concerning the insertion procedure and the coding of video data are taken into account. As an example, the complexity of the insertion algorithm can be greatly simplified by using the transparency mask to provide useful information about the position of the ISL. Hence, every macroblock (MB) whose elements of the corresponding transparency mask $m(n_1, n_2)$

are all equal to zero ($\forall (n_1, n_2) \in \text{MB}_x \ m(n_1, n_2) = 0$) can be skipped from the whole insertion procedure.

More than the desired position of the logo or subtitle, it is also convenient to know the width and the height of the hypothetical box that bounds the ISL tightly. Some authors have suggested that, whenever it is possible, the position of this box should be adjusted so that the number of MBs affected by the ISL insertion is minimized [8].

One other important issue is concerned with the effect of the insertion on the coding of the several MBs. In fact, despite all the strategies that have been proposed by several authors over the last few years to re-use as much information decoded from the incoming bit-stream as possible in the coding of the output bit-stream, one now has to take into account that this information may no longer be valid for the MBs in the vicinity of the insertion. In fact, considering that video transcoders usually re-use the motion vectors (MVs) decoded from the input bit-stream, one now has to take into account that these MVs may no longer point to the best matching MB. Furthermore, it is also possible that MVs that pointed to the MBs where the logo has been inserted may have to be re-estimated, since the logo content is not expected to be inserted in those regions.

Panusopone [8] has recently studied the problem of inserting translucent logos (see eq. 1) in pixel domain transcoders and proposed several schemes to adapt the motions vectors and the quantization steps so that the impact on the coding algorithm is small [8]. Despite the fact that the used insertion techniques are clearly different from those presented in this paper and the fact that he has proposed an insertion method for the pixel domain, his proposals concerning the re-usage of the MVs can still be applied to the scheme presented above with minor changes.

5. EXPERIMENTAL RESULTS

The proposed insertion method in the compressed DCT domain was used to insert two ISLs, corresponding to the logo presented in fig. 4(a) and the subtitle presented in fig. 4(b), in the CIF *carphone* video sequence at positions (241, 10) and (256, 130), respectively. These posi-

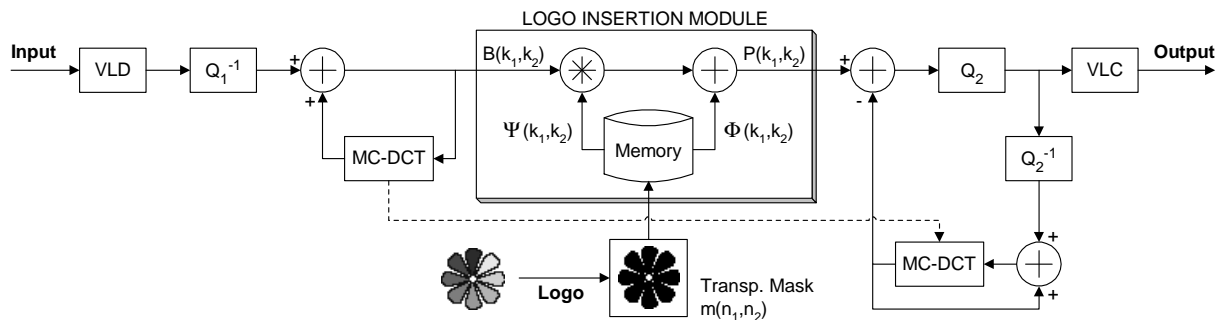


Figure 3. Block diagram of the compressed domain transcoder with a logo insertion module.



Figure 5. Logo insertion in the compressed domain using the *carphone* sequence and the transparency mask of fig 5(c).

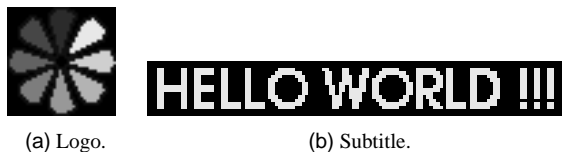


Figure 4. Considered set of ISL's ($T = 0$).

tions were carefully selected in order to affect the minimum number of MBs as possible.

The frame presented in fig. 5(a) was obtained using a transparency factor of $\alpha = 0.5$. As it can be seen, it is possible to note the presence of some details of the background image in the area corresponding to the inserted logo. Fig. 5(b) presents the same frame obtained using a transparency factor of $\alpha = 1.0$. Contrasting with the previous setup, in this case the insertion is 100% opaque. The corresponding transparency mask $m(n_1, n_2)$ is presented in fig. 5(c).

6. CONCLUSIONS

A logo insertion scheme in the compressed DCT domain for video transcoding was presented. This scheme was adapted from the well-known pixel domain compositing technique, so that only the pixels corresponding to those objects that are intended to be inserted are considered in the operation. By using this scheme, the problem concerning the presence of an undesired semi-transparent region around irregular-shaped objects (such as logos or subtitles) was circumvented, by using a corresponding transparency mask. This technique implied the usage of a symmetric convolution operator in the compressed DCT domain. The inclusion of a logo insertion module in the architecture of a compressed domain video transcoder was also discussed.

REFERENCES

- [1] Jianhao Meng and S.-F. Chang, "Embedding visible video watermarks in the compressed domain," *Proc. of IEEE International Conference on Image Processing (ICIP'1998)*, volume 1, pp. 474–477, 1998.
- [2] P. Assunção and M. Ghanbari, "A frequency-domain video transcoder for dynamic bit-rate reduction of MPEG-2 bitstreams," *IEEE Transactions on Circuits and Systems for Video Technology*, volume 8, no. 8, pp. 953–967, December 1998.
- [3] Bo Shen and K. Sethi, "Block-based manipulations of transformed-compressed images and videos," *Multimedia Systems Journal*, volume 6, no. 2, pp. 113–124, March 1998.
- [4] T. Porter and T. Duff, "Compositing digital images," *Computer Graphics (Proc. SIGGRAPH'84)*, volume 18, no. 3, pp. 253–259, July 1984.
- [5] B. C. Smith and L. A. Rowe, "Algorithms for manipulating compressed images," *IEEE Computer Graphics & Applications*, pp. 34–42, September 1993.
- [6] S. Martucci, "Symmetric convolution and discrete sine and cosine transforms," *IEEE Transactions on Signal Processing*, volume SP-42, no. 5, pp. 1038–1051, May 1994.
- [7] S.-F. Chang and D. G. Messerschmitt, "Manipulation and compositing of MC-DCT compressed video," *IEEE Journal Selected Areas in Communications*, volume 13, no. 1, pp. 1–11, January 1995.
- [8] K. Panusopone, X. Chen and F. Ling, "Logo insertion in MPEG transcoder," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, Salt Lake City - USA, May 2001.