# Compressed Domain Video Resolution Enhancement: Use of Motion Vectors

Narasimha Kaulgud
E and C. Dept.
S.J.College of Engg.
Mysore, India
kaulgud@sjce.ac.in

Jayahsree Karlekar
Hughes Software
Gurgoan
New Delhi, India
jkarlekar@hss.hns.com

U. B. Desai
Electrical Engg. Dept.
IIT–Bombay
Mumbai, India
ubdesai@ee.iitb.ac.in

## Abstract

*In this paper we propose a new method for video resolution enhancement from compressed video data. Novelty of the approach is that resolution enhancement is done by interpolating the motion vectors. Proposed method works very well for different compression methodologies. We first discuss zooming, when the video is compressed using the Discrete Cosine Transform Domain (DCT). Next we use the Discrete Wavelet Transform (DWT) based video compression proposed in [7] and develop a zooming technique in the DWT domain. Simulation results are presented to illustrate the efficacy of the proposed method. Finally, we show that motion vector interpolation can also be used from video frame interpolation.*

**Key words:** Video, Zooming, Motion Vectors, Compressed Video, Wavelets, DCT, Frame interpolation.

## 1 Introduction

Due to limitations in the channel capacity, a low resolution video is transmitted in the compressed domain. Typically, motion information from the adjacent frames - which exploits temporal redundancy - is coded and quantized for video compression. Of course, spatial redundancy is handled using a transform domain encoding. Conventional method is to first decode the low resolution video and then apply an interpolation technique to obtain the zoomed video. This approach is not only computationally expensive, but may result in poor video quality. On the other hand, if we can interpolate the data in the compressed domain, then we can achieve better decoded video quality at a lower computational cost. Our approach focuses on developing a method for interpolating the transform domain coefficients and the motion vectors to zoom a compressed video stream.

Video zooming in non-compressed domain is overviewed by Katsaggelos and Galatsanos [9]. Of late, motion zooming in compressed domain is attracting a lot of attention; for example, Seagul and Katsaggelos [18] propose visual quality measurement constraint for enhancement of compressed video, and Mateos *et al* [15] propose a scheme for simultaneous motion estimation and resolution enhancement. In one of our earlier papers (Kaulgud and Desai [12]) we had given some preliminary results on video zooming in compressed domain.

This paper is organized as follows: Section 2 of the paper, very briefly, reviews a few concepts of video compression and motion estimation methodologies. Section 3 discusses the proposed technique of video zooming in the compressed domain. Section 4 extends the ideas from section 3 for temporal video zooming (frame interpolation) and finally section 5 presents the experimental results.

## 2 Video Compression Technique

A large number of motion estimation algorithms have been proposed in literature. A good review of motion estimation is presented by Aggarwal and Nandhakumar [1]. Comparison of different motion estimation techniques are compiled by Dufuax *et al* [4] and Hand [6]. The book by Konstanides and Bhaskaran [2] provides an excellent treatment on DCT based video compression techniques (for example, H.263, MPEG-1, MPEG-2, etc). Thus we forego the details.

### 2.1 Multiresolution Techniques

In the DWT domain, normally, motion compensation is done in the spatial domain on sub-blocks of size $8 \times 8$ or $16 \times 16$ which necessarily means reconstruction (inverse transform) of the previously encoded reference frame for motion estimation. To avoid computing IDWT, Zhang and Zafar [21] proposed a multiresolution motion estimation (MRME), which estimates motion in wavelet transform domain. This technique estimates the motion vector hierarchically form lower to higher resolution sub-images. In MRME scheme, the motion vectors for the detail sub-image of the coarsest resolution are determined (level-3 of Figure 1) using the conventional block matching based motion estimation algorithm. These motion vectors are used as the initial bases for calculating the motion vectors at the next finer resolution. Errors are likely to occur and propagate to the other levels of detailed sub-images. Various methods have been proposed

to overcome this problem (see for example [13, 17, 20]). A modification of the MRME method is was proposed in the earlier work [8].

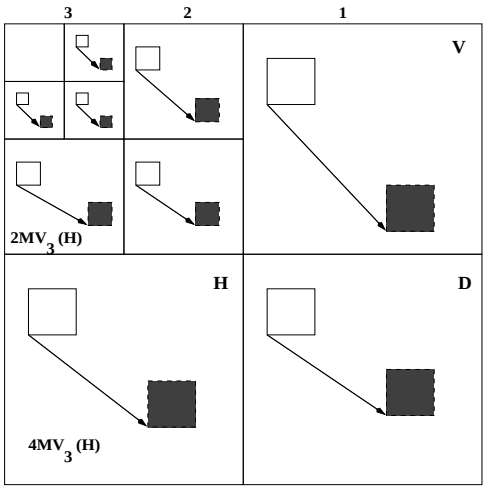


Figure 1: 3-level wavelet decomposition of a video frame. Empty blocks: position of block in current frame, shaded blocks:position of the block used for motion compensation from previous frame. Motion vectors obtained at level-3 are propagated to do motion compensation at level 2 and 1

Motion vectors at level 3 are first computed then propagated to levels 1 and 2. This is done assuming that the motion vectors at different levels are highly correlated[1]. Based on this, Karlekar and Desai [7] (see also Kaulgud [10]) proposed a new DWT based video encoding technique depicted in Figure 2.

# 3 Proposed Compressed Domain Video Zooming

We assume that the motion vectors have been extracted from the compressed bitstream. With this assumption, task on hand is to interpolate these motion vectors. We shall describe the proposed scheme for the DCT domain and then for DWT domain.

**DCT based Scheme:** Let ${}^k b_p^8$ be the $k^{th}$ sub-block of $p^{th}$ frame of a given video sequence in the spatial domain. Superscript 8 denotes size of the sub-block, $8 \times 8$, in this case. Correspondingly ${}^k b_{p-1}^8$ denotes the sub-block in the $(p-1)^{th}$ frame. Our aim is to generate ${}^k \hat{b}_p^{16}$, an estimate of the zoomed $16 \times 16$ sub-block from these two blocks. Let the DCT of these two blocks be ${}^k B_p^8$ and ${}^k B_{p-1}^8$, respectively. To estimate the motion vector in the neighborhood $\Omega$, a block matching method based on minimum of absolute difference (MAD) is used, namely,

$$
\begin{aligned}
&{}^k v = ({}^k v_x, {}^k v_y) = \\
&\arg\min_{(i,j)\in\Omega} \sum_{m=0}^{7} \sum_{n=0}^{7} [|{}^k b_p^8(x+m, y+n) - \\
&{}^k b_{p-1}^8(x+i+m, y+j+n)|]
\end{aligned}
\tag{1}
$$

where $(x, y)$ is the upper left corner coordinates of the $p^{th}$ block. Next let

$${}^k \epsilon_p = |\sum_i \sum_j \{{}^k B_p^8(i,j) - {}^k B_{p-1}^8(i,j)\}|$$

where $(i, j)$ give the the transform domain coordinates for the DCT of the sub-block $p$ and the DCT of the motion compensated sub-block $p-1$. We assume ${}^k v$ and ${}^k \epsilon_p$ are available. The task on hand now is to extrapolate these motion vectors from ${}^k v$ to ${}^k \hat{v}$ and use it to zoom the video. Based on extensive empirical study, we found that a linear extrapolation scheme works well, namely, ${}^k \hat{v}_x = 2({}^k v_x)$ and ${}^k \hat{v}_y = 2({}^k v_y)$. The new block locations are now evaluated as

$${}^k \hat{B}_p^8(i,j) = ({}^k B_{p-1}^8(i + {}^k \hat{v}_x, j + {}^k \hat{v}_y) + {}^k \epsilon_p) \tag{2}$$

It is shown in [3, 16] that zero padding provides a good approximation to the high resolution (zoomed) DCT. We use the same idea for zooming the $8 \times 8$ DCT block ${}^k \hat{B}_p^8$ to size $16 \times 16$. Let

$${}^k \hat{B}_p^{16} = \begin{bmatrix} {}^k \hat{B}_p^8 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{3}$$

where, $\mathbf{0}$ is an $8 \times 8$ null matrix. Inverse of ${}^k \hat{B}_p^{16}$ gives ${}^k \hat{b}_p^{16}$ or a $16 \times 16$ spatial domain block. In effect, we obtain a video frame zoomed by a factor of two in the $x$ and $y$ directions.

**DWT based Scheme:** Unlike DCT, DWT has a nice multiresolution structure which can be exploited for estimating the wavelet coefficients at finer scales (higher resolutions). We replace $\mathbf{0}$ of equation (3) by *estimated* wavelet coefficients.

$${}^k B_p^{16} = \begin{bmatrix} {}^k \hat{B}_p^8 & {}^k \hat{B}_p^8(V) \\ {}^k \hat{B}_p^8(H) & {}^k \hat{B}_p^8(D) \end{bmatrix} \tag{4}$$

where, ${}^k \hat{B}_p^8(V)$ correspond to the estimated wavelet coefficients in the vertical direction. Similarly, H and D for horizontal and diagonal directions, respectively. We next describe the procedure to estimate these wavelet coefficients.

We have proposed a new method to estimate the wavelet coefficients [11]. To illustrate the estimation of the coefficients, consider Figure 3.

We assume that the DWT of an $M \times M$ video frame composed of boxes $0, I, II, IV, V, VII, VIII$, is available and we want to zoom it to size $2M \times 2M$. This would be possible if we can estimate the wavelet coefficients in boxes $III, VI$ and $IX$. Having estimated these wavelet coefficients, we simply feed these along with the $M \times M$

[1]This assumption is basis for our proposed video zooming

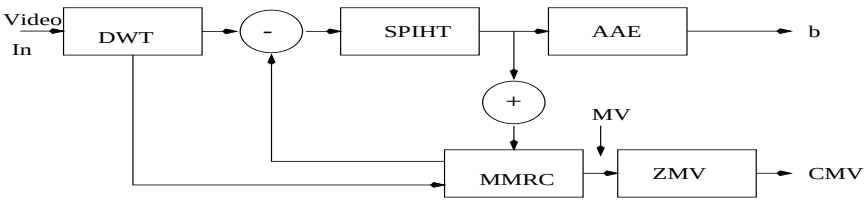


Figure 2: DWT Based Video Encoder: DWT–Discrete Wavelet Transform, SPHIT–Set Partitioning in Hierarchical Trees, MRMC–Multiresolution Motion Compensation, AAE–Adaptive Arithmetic Encoder, b–Output Bits, MV–Motion Vectors, CMV–Coded Motion Vectors, ZMVC–Zerotree Motion Vector Coder

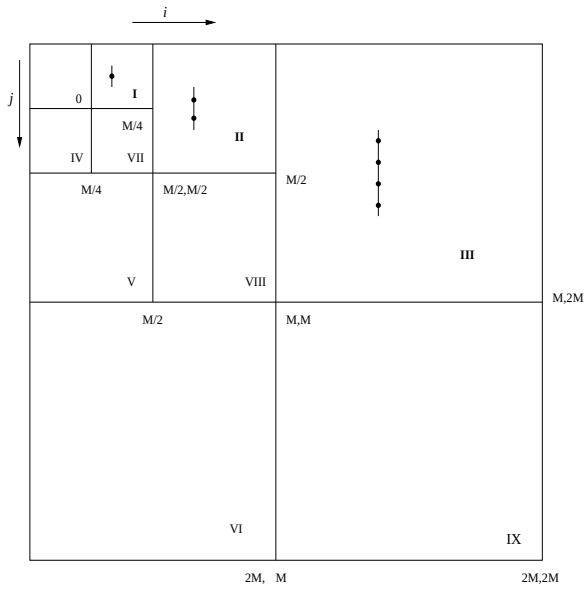


Figure 3: Zooming Process

video frame to the wavelet based video frame synthesis filter bank and obtain the interpolated (zoomed) video frame of size $2M \times 2M$. To estimate the wavelet coefficients in boxes *III*, *VI* and *IX*, we find the significant wavelet coefficients at two resolutions (namely, in boxes $I - II$, $IV - V$, $VII$ - $VIII$). For example, consider boxes $I$ and $II$ of Fig. 3, with significant coefficients shown as dots. Denote coefficients in respective boxes as: $d_1(i_1, j_1) \in I$ and $d_2(i_2, j_2) \in II$. Note that, $i_1, j_1$ satisfy $M/4 \leq i_1 \leq (M/2)-1$ and $0 \leq j_1 \leq (M/4)-1$. Also, $i_1$ and $i_2$ are related by $i_1 = \lfloor i_2/2 \rfloor$ ($\lfloor . \rfloor$ represents the floor operator); $j_1$ and $j_2$ are similarly related. We define $D_{(.)}(i, j)$ as (between boxes $I$ and $II$):

$$D_1(i,j) = \frac{d_2(i,j)}{d_1(\lfloor i/2 \rfloor, \lfloor j/2 \rfloor)} \tag{5}$$

$$D_2(i,j) = \frac{d_2(i,j+1))}{d_1(\lfloor i/2 \rfloor, \lfloor (j+1)/2 \rfloor)} \tag{6}$$

These $D_{(.)}(i, j)$ values are used to estimate coefficients $\hat{d}$ at the finer scale (box *III*).

$$\begin{aligned} \hat{d}(2i, 2j) &= D_1(i,j) d_2(i,j)(1 - l_{d(i,j)}) \\ \hat{d}(2i, 2j+2) &= D_2(i,j) d_2(i,j+1)(1 - l_{d(i,j+1)}) \end{aligned} \tag{7}$$

We set:

$$\hat{d}(2i, 2j+1) = \hat{d}(2i, 2j), \quad \hat{d}(2i, 2j+3) = \hat{d}(2i, 2j+2) \tag{8}$$

$l_{d(i,j)}$ is an indicator function; $l_{d(i,j)}$ is set to zero, if $d(i, j)$ is significant, else to one. We define $d(i, j)$ to be significant if $|d(i, j)| > \text{threshold}$. Note that (7) implies an exponential decay and this is consistent with what is reported in [5, 14, 19]. Thus, we refer to $D_{(.)}(i, j)$ as the decay parameter. In principle, for each coefficient $d_2(i, j) \in II$ we should have four coefficients in box *III*. But, our experiments have shown that doing this leads to a "blocky" zoomed video frame Hence, we generate only *two* coefficients in box *III* corresponding to $d_2(i, j) \in II$. Moreover, we know that the detail sub-images using the wavelet transform yields vertical lines in boxes $I$, $II$ and $III$; horizontal lines in boxes $IV$, $V$ and $VI$; and diagonal lines in boxes $VII$, $VIII$ and $IX$. We use this intuition and compute wavelet coefficients along vertical direction in box $III$, along horizontal direction in box $VI$, and along diagonal direction in box $IX$ and that too only along alternate lines. For box *III* equation (7) and (8) hold good. Analogous expressions can easily be obtained for wavelet coefficient in box $IV$ and $IX$. Now, the estimated $\hat{d}$'s and the original $M \times M$ video frame is fed to the wavelet based video frame synthesizer to obtain the zoomed video frame which is of twice the size of the given video frame In all our simulation the threshold was selected as half the maximum coefficient in the respective boxes, namely boxes $II$, $V$ and $VII$.

In the DWT based video encoder, the motion vectors are computed in the DWT domain. Thus, we apply the same process as above to interpolate the motion vectors. Motion compensation is applied in the DWT domain using the interpolated motion vectors. An inverse DWT on the motion compensated zoomed DWT blocks will lead to the zoomed video.

# 4 Temporal zooming

Motion vector interpolated video frame zooming method can be extended interpolating temporal video frames. This involves estimating *missing* frames in a given video stream. This is achieved by developing a DWT based method for temporal interpolation of motion vectors. For

comparison, we also implement a linear interpolation scheme. Suppose, we have two frames $p-1$ and $p+1$, and we need to estimate the *missing* frame $p$. As in the previous section, let ${}^k b^{16}_{p+1}$ and ${}^k b^{16}_{p-1}$ be the $k^{th}$ macroblock of frames $p+1$ and $p-1$ respectively. Let ${}^k B^{16}_{p+1}$ and ${}^k B^{16}_{p-1}$ be their respective DWTs. The MAD is now defined as

$$\begin{array}{l} {}^k w = ({}^k w_x, {}^k w_y) = \underset{(i,j)\in\Omega}{arg\ min} \\ \sum_{m=0}^{15}\sum_{n=0}^{15} |{}^k b^{16}_{p+1}(x+m, y+n) - \\ {}^k b^{16}_{p-1}(x+i+m, y+j+n)| \end{array} \quad (9)$$

Error is defined as,

$${}^k \epsilon_p = |\sum_{i=0}^{15}\sum_{j=0}^{15}\{{}^k B^{16}_{p+1}(i,j) - {}^k B^{16}_{p-1}(i,j)\}| \quad (10)$$

Motion vectors for the $p^{th}$ frame are estimated as ${}^k \hat{w}_x = \lfloor \frac{{}^k w_x}{2} \rfloor$ and ${}^k \hat{w}_y = \lfloor \frac{{}^k w_y}{2} \rfloor$. The new block location in the $p^{th}$ frame will be,

$${}^k \hat{B}^{16}_p = ({}^k B^{16}_{p-1}({}^k \hat{w}_x . x + i, {}^k \hat{w}_y . y + j) + {}^k \epsilon_p \quad (11)$$

Since the size of the estimated video frame $p$ is the same as the available frames viz., $p-1$ and $p+1$, we do not need Eqn.s (3 and 4).

# 5 Results and Discussions

We compare the frame interpolation scheme with linear frame interpolation, where a pixel at location $(i,j)$ of frame $p$ is the average of pixels at the same $(i,j)$ locations in frame $p-1$ and $p+1$. Note that the linear frame interpolation is done in the *spatial* domain and not in the compressed domain. Performance is compared using SNR defined $SNR = 10\ \frac{log \sum_{i,j} X^2_i, j}{\sum_{i,j}(X_{i,j} - \hat{X}_{i,j})^2}$. $X$ being the original frame and $\hat{X}$ being the estimated frame. Plots of this method are shown in Fig.s 4 - 5 for the Claire video. We see that visually there is not much difference between linear and motion vector (MV) interpolated methods. But the PSNR plots show that linear interpolation is slightly better than the proposed method. This is due to the fact that MV based scheme is carried out in compressed domain. During MV estimation, coding and quantization errors may have crept in. However, the difference is small, and MV based scheme is comparable, if not better, to the linear interpolation scheme. Here, we note that for SNR computation, original frame is available, unlike the zooming application.

For SNR computation original frame is not available. So, we generate a zoomed video sequence without using the motion vectors in the DCT framework. This represents the worst case scenario, We use this video (represented as $\tilde{X}$) as the reference and the zoomed video $\hat{X}$ obtained using the DCT or DWT scheme as the output

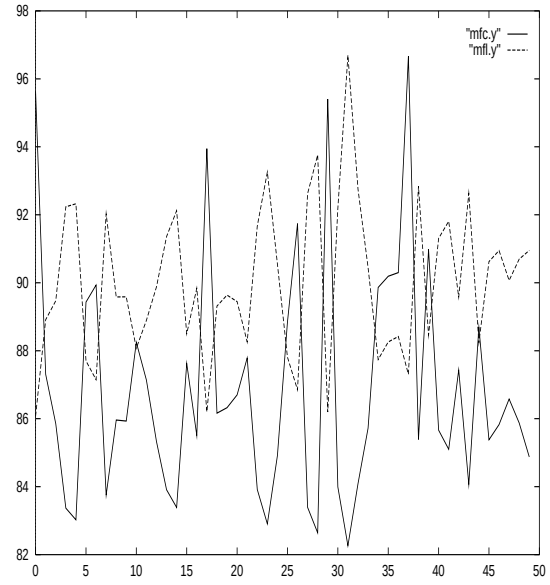


Figure 4: SNR plot for Y component of frame interpolated Claire video. Thick lines for DWT based interpolation and thin lines for linear frame interpolation. $Y$ axis is the SNR and $X$ axis is the frame No.

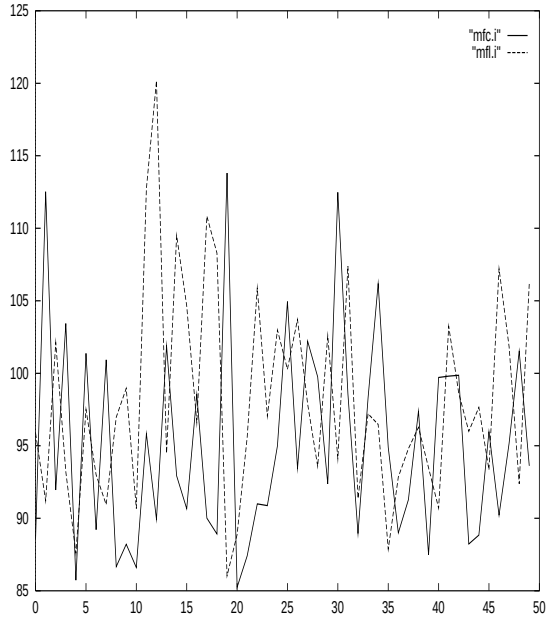


Figure 5: SNR plot for I component of frame interpolated Miss America video. Thick lines for DWT based interpolation and thin lines for linear frame interpolation. $Y$ axis is the SNR and $X$ axis is the frame No.

zoomed video. Signal to Noise Ratio (SNR) is measured for each frame $p$ as $SNR_p = 10\ log\ \frac{\sum_{i,j} \tilde{X}_{i,j}^2}{\sum_{i,j}(\tilde{X}_{i,j} - \hat{X}_{i,j})^2}$ Figure 6 show the $Y$ component of SNR plots for Suzie video. The plots and resulting video (see sample clips in Figure 7) suggests that the proposed method for video zooming works quite well. In particular, as one would expect, the DWT based scheme works better.

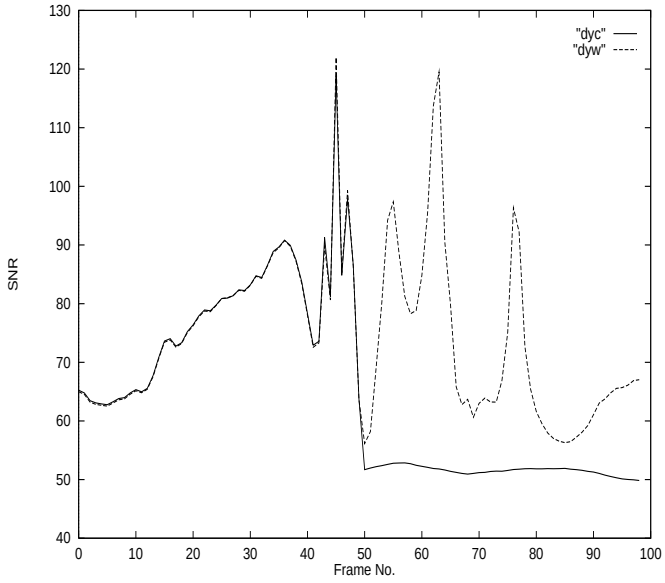


Figure 6: SNR plot for Y component of Suzie video. Thick lines are for DCT and dotted lines are for DWT

Sample video frames are shown in Fig. (7). We have shown frames 3-6 of Suzie video. Left column shows the original video frames, middle shows the results of DCT based scheme and the right column shows the results of the DWT based scheme.

We would like to mention that we have considered QCIF video clips [2] for our experiment; but the technique should work satisfactorily for other video formats as well.

# References


[1] J. K. Aggarwal and N. Nandhakumar. "On computation of motion from sequence of images- A review". *Proc. of IEEE*, 76:917–935, Aug 1988.

[2] V. Bhaskaran and K. Konstantinides. *"Image and video compression standards, algorithms and architectures "*. Kluwer Academic, Boston, 1998.

[3] Bo Shen, Ishwar Sethi and Bhaskar Vasudevan. "Adaptive motion vector resampling for compressed video downsampling ". *IEEE Tx. on Ckts and systs for video technology*, 9(6):929–936, Sept. 1999.

[4] F. Dufaux and F. Moscheni. "Motion estimatin techniques for digital TV: A review and a new contribution". *Proc. of IEEE*, 83:856–876, June 1992.


[2] sample video clips are available at http://www.geocities.com/kaulgud/iasted


[5] G. Grace Chang, Zoran Cvetkovicand and Martin vetterli. "Resolution enhancement of image using wavelet transfomr extrema intepolation". In *"IEEE ICASSP"*, pages 2379–2383, 1995.

[6] Hseuh-Ming Hang and John W Woods. *"Hand book of visual communications"*. Acadenic Press, New York, 1995.

[7] Jayashree Karlekar and U. B. Desai. "SPIHT video coder". In *"Proceedings of IEEE TENCON-98 , New Delhi"*, 1998.

[8] Jayashree Karlekar and U. B. Desai. "New multiresolution motion estimateion and comprensation scheme". In *"IEEE Intnl. Conf. on ckts and sys."*, 1999.

[9] A. K. Katsaggelos and N. P. Galastanos. *" Signal recovery techniques for image and video compression and transmissiion "*. Kluwer Academic, 1998.

[10] Narasimha Kaulgud. *"New approaches to color image restorationa and zooming of compressed video "*. PhD thesis, IIT Bombay, Electrical Engg. Dept., 2002.

[11] Narasimha Kaulgud and U. B. Desai. *"Image zooming:Use of wavelets"*. in "Superresolution Imaging", Ed. Subhasis Chaudhuri, Kluwer-Academic, 2001.

[12] Narasimha Kaulgud and U. B. Desai. "Video magnification in comppressed domain". In *"Proceedings of Natnl. Conf. on Communications NCC-2001, Kanpur"*, 2001.

[13] M. K. Mandal, E. Chan, X. Wong and S. Panchanathalu. "Multiresolution motion estimation techniques for video compression". *Opt. Engg.*, 35:128–136, 1996.

[14] Stephen Mallat and Stefan Zhong. "Characterization of singnals from multiscale edges ". *IEEE PAMI*, 14:700–732, 1992.

[15] Javier Mateos and A. K. Katsagelos. "Simultaneous motion estimation and resolution enhancement of compressed low resolution video", 2000.

[16] B. Natarajan and V. Bhaskaran. "A fast approximate algorithm for scaling dwon digital images in the DCT domain". In *"IEEE-ICASSP "*, pages 2307–2310, 1995.

[17] S. Kim, S. Rhee, J. G. Jeon and K. T. Park. "Interframe coding using two stage variable block size mltiresolution motion estimation and compression". *IEEE Tx. on Circuits and systems for Video Tech.*, 8:399–409, Aug 1998.

[18] C. Andrew Seagal and A. K. Katsagelos. "Enhancement of compressed video using visual quality measurement", 2000.

[19] W. Knox Carey, Daniel Chuang and Sheila Hemami. "Regularity preserving image interpolation ". *IEEE Tx on Image Processing*, 8(9):1293–1297, Sept. 1999.

[20] Z. Zafar Y. Q. Zhang and B. Jabbari. "Multiscale video representation usnign multiresolution motion compensation and wavelet decomposition". *IEEE Tx. on selected areas in Commn.*, 11:24–35, Jan. 1993.

[21] Y. Q. Zhang and S. Zafar. "Motion compensated wavelet transform coding". *IEEE Tx on Circuuits and systems for Video Tech.*, 2:285–296, Sept. 1992.

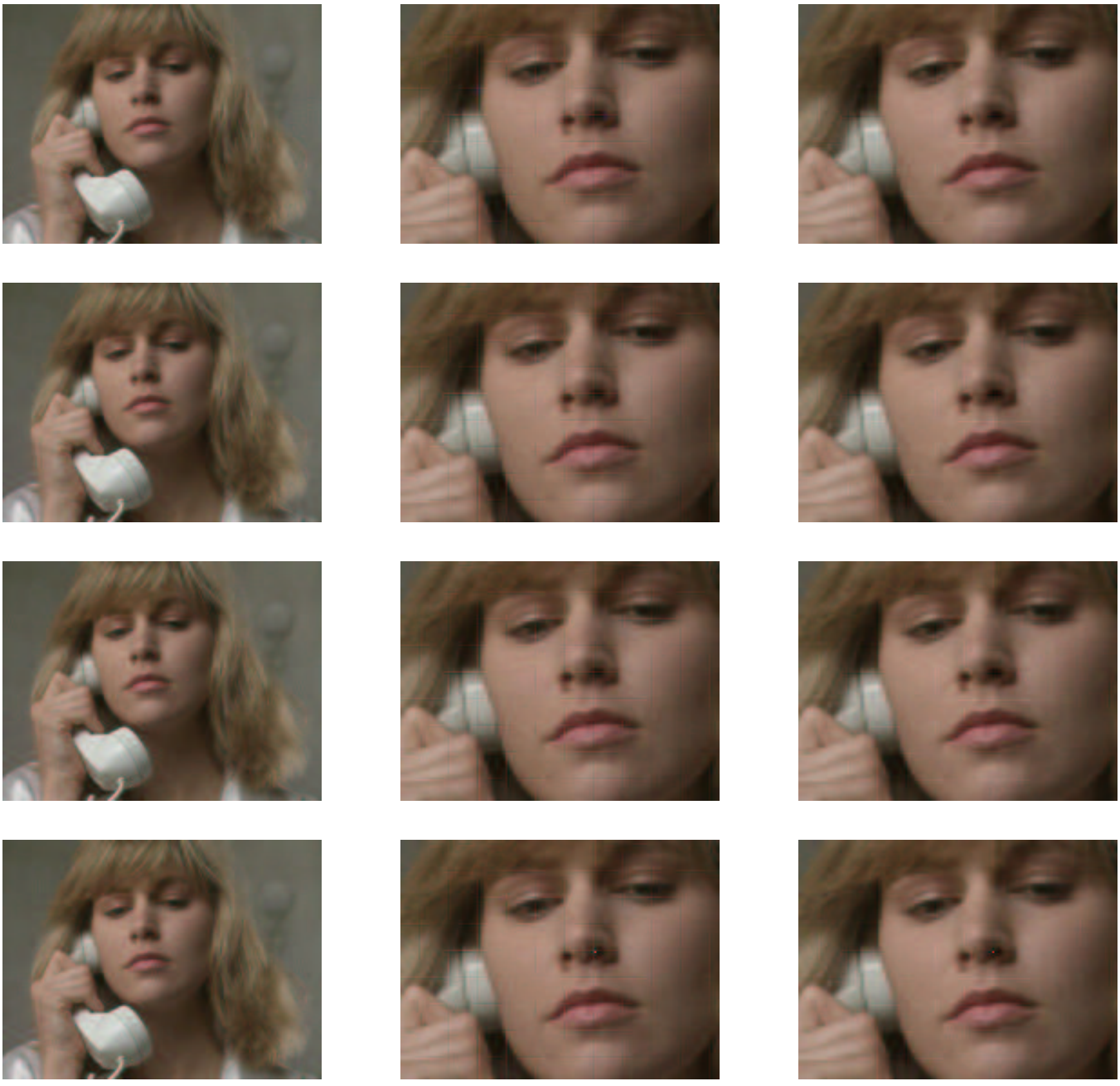

Figure 7: Frames 3-6 (top to bottom) of Suzie image: (left) original frames, (middle) Interpolated using motion vectors in DCT domain, and (right) in DWT domain