Abstract—In this paper, we present an adaptive directional window selection for the edge-directed interpolation. The new window selection can solve the problem of covariance mismatch in high frequency and texture regions. It makes use of a practical directional elliptic window which works according to the edge direction sliding along an edge and then subsequently chooses the best window evaluated by choosing the elliptic window which has the lowest Means Square Error (MSE). Experimental results show that by the proposed technique can generate a high quality interpolated image which is better than other edge directed interpolation approaches. Experimental results also provided on different images to justify the value of this approach at the end of the paper.

Keywords-component; Interpolation, edge-directed, sample window selection

I. INTRODUCTION

High Definition television (or HDTV) can produce a better visual quality than Standard Definition (or SDTV). It is because HDTV has a new digital television broadcasting system with a high-resolution display. However, not all the videos or movies produced with the high-resolution quality. Many videos or movies only contain a standard resolution (720×576) video sequence. Therefore, interpolation algorithms are demanding, such that standard resolution video sequence can be shown on high-resolution display units. It is well know that using a classical linear interpolation algorithm, such as bilinear and bicubic interpolation cannot produce a visual quality that is accepted by today customer, as it often suffers from the edge blurring effect or produce some artifacts around the edge area [1]. These artifacts will reduce substantially the visual quality of the video sequence, especially the artifacts that appear near the major edges. Therefore, many research studies [2]-[11] tried to improve the visual quality and lower the mean-squares errors and to make comparison with the linear interpolation algorithm.

The New Edge-Directed Interpolation (NEDI) [12] method models natural images as a second-order locally stationary Gaussian process and predicts unknown pixels. The NEDI method uses a square window which is located at the center of the unknown pixels in order to model the statistics near the unknown pixels, so that the unknown pixels can be predicted. Recently, a Modified Edge-directed interpolation MEDI [13] was proposed in 2009, which suggested to use a multiple square training windows instead of a single square training window. However, we find that using multi-square windows do not always get the optimal result especially in the fine edge and texture regions. Therefore, we propose in this paper an Adaptive Directional Window Selection to overcome the exiting problem in the Edge-Directed Interpolation (EDI).

II. PREVIOUS WORK

In this section, let us review the basic principle of NEDI [12] and it’s window selection method.

A. Basic Principle of NEDI

Consider the interpolation of an image $X$ with size $H \times W$ to a high-resolution image $Y$ with size $2H \times 2W$. The white dots in Fig.1 denote the original low-resolution pixels $Y_{2i,2j}$ and the gray dots denote the unknown pixel $Y_{2i,2j+1}$ which is to be estimated by the NEDI step one. The NEDI step one makes use of a fourth-order linear prediction to interpolate the unknown pixels $Y_{2i+1,2j+1}$ from the four neighboring pixels \{$Y_{2i,2j} , Y_{2i+2,2j} , Y_{2i+2,2j+2}, Y_{2i+2,2j+2}$\} as shown below:

$$Y_{2i+1,2j+1} = \sum_{k=0}^{1} \sum_{l=0}^{1} \alpha_{2k+1,2l+1} Y_{2i+k,2j+l+1}$$  \hspace{1cm} (1)

According to Wiener filtering theory, the optimal Minimum Means Square Error (MMSE) prediction coefficients set $\alpha$ can be obtained as [12]

$$\alpha = R_{yy}^{-1} r_y$$ \hspace{1cm} (2)

where $\alpha = \{\alpha_0, ..., \alpha_6\}$, the auto-covariance $R_{yy}$, is a square matrix containing sixteen $R_{ii}$ with $k, l = \{0, ..., 3\}$ for the position in the sample points set, and the cross-covariance of $r$, contains four $r_i$ for $l=\{0, ..., 3\}$. For example, $r_0$ is defined by $E[Y_{2i,3}Y_{2i+2,2j+1}]$ and $R_{02}$ is defined by $E[Y_{2i,2}Y_{2i+2,2j+2}]$ as shown in Fig. 1.

The high-resolution cross-covariance $r_y$ is not available, because of the center pixel $Y_{2i+1,2j+1}$ is to be predicted. This difficulty can be overcome by the fact that the statistics of the pixels with respect to the low-resolution block and that of the high-resolution block are most likely to be similar. As a result, the auto-covariance and cross-covariance coefficients among the high-resolution block will be mostly alike that of the low-resolution block. Therefore, the low-resolution covariance $R_{yy}$ and $r_y$ will be used instead, for the calculation.
According to the classical covariance method \([12]\), \(R_{y}^{\prime}y\) and \(r_{y}\) can be calculated by the following equation
\[
R = \frac{1}{M} C^T C, \quad r = \frac{1}{M} C^T y
\]  
(3)

where \(y = [y_1, y_2, ..., y_{M\times M}]^T\) is a data vector containing \(M\times M\) pixels inside a local window and \(C\) is a \(M^2\times4\) matrix,
\[
C = \begin{bmatrix}
C_{1,1} & C_{1,2} & C_{1,3} & C_{1,4} \\
\vdots & \vdots & \vdots & \vdots \\
C_{M\times M,1} & C_{M\times M,2} & C_{M\times M,3} & C_{M\times M,4}
\end{bmatrix}
\]

whose \(k^{th}\) row vector contains four nearest neighbours of \(y_k\) in the diagonal direction, i.e. if \(y_k = Y_{2i,2j}\), then the four nearest neighbours are \(\{Y_{2i,2j-2}, Y_{2i,2j+2}, Y_{2i+2,2j-2}, Y_{2i+2,2j+2}\}\) as shown in Fig. 1. If \(y_k = y_{1,1}\), then the four nearest neighbors are \(\{C_{1,1}, C_{1,2}, C_{1,3}, C_{1,4}\}\). When the local window is \(4\times4\), vector \(y\) and matrix \(C\) can be find according to the Fig. 2.

According to (2) and (3), we have
\[
\alpha = (C^T C)^{-1}(C^T y)
\]  
(4)

After the calculation of filter coefficients of \(\alpha\), the interpolated value of \(Y_{2i+1,2j+1}\) can be then obtained by substituting (4) into (1).

### B. Original Window Selection Method

The original window selection method in NEDI algorithm is simply and it just uses a square window, with its center at the unknown pixel that we want to predict as shown in Fig. 2. The assumption of using a square window is that the statistics of this square window in the low-resolution domain should be the same as the statistics in high-resolution domain. However, in many cases this assumption is not true. Therefore, the NEDI always introduces directional artifacts, as the statistics of the data in the square window cannot truly reflect the real situation.

### III. PROPOSED ADAPTIVE DIRECTIONAL WINDOW SELECTION (ADWS)

Our proposed ADWS method includes three steps. First, Edge Direction Detection is applied to the image. Second, the Adaptive Directional Elliptic Window is then also applied according to the edge Direction. Finally, in the Window Selection step the best window is selected for forming the Wiener filter.

#### A. Edge Direction Detection

As shown in Fig. 3, there are three pixels to be interpolated \((D, H, and V)\) with interpolation factor equal to 2. Let us classify the edge direction into nine categories (including a small-edge type and eight edge types with directions: \(0^\circ, 22.5^\circ, 45^\circ, 67.5^\circ, 90^\circ, 112.5^\circ, 135^\circ, 157.5^\circ\)) in the diagonal \((D)\), horizontal \((H)\) and vertical \((V)\) edge detections. We modified the edge detection method in \([14]\) to reduce the computational cost.

In this study, the eight filtering masks for diagonal edge detection of eight directions are as follows:

\[
D_{y} = \begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & -1 & -1 & -1 \\
0 & 1 & 1 & 1 \\
0 & 0 & 1 & 0
\end{bmatrix}, \quad D_{x} = \begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & -1 & -1 & -1 \\
0 & 1 & 1 & 1 \\
0 & 0 & 1 & 0
\end{bmatrix}
\]

\[
D_{z} = \begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & -1 & -1 & -1 \\
0 & 1 & 1 & 1 \\
0 & 0 & 1 & 0
\end{bmatrix}, \quad D_{w} = \begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & -1 & -1 & -1 \\
0 & 1 & 1 & 1 \\
0 & 0 & 1 & 0
\end{bmatrix}
\]

\[
D_{z} = \begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & -1 & -1 & -1 \\
0 & 1 & 1 & 1 \\
0 & 0 & 1 & 0
\end{bmatrix}, \quad D_{w} = \begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & -1 & -1 & -1 \\
0 & 1 & 1 & 1 \\
0 & 0 & 1 & 0
\end{bmatrix}
\]  
(5)
The eight filtering masks for horizontal edge detection of eight directions are as follows:

\[
H_{x} = \begin{bmatrix}
0 & 0 & -1 & -1 & 0 \\
0 & -1 & -1 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 1 \\
0 & 0 & 1 & 1 & 0 \\
0 & 0 & -1 & -1 & 0 \\
0 & -1 & -1 & -1 & 0 \\
0 & -1 & -1 & -1 & 0
\end{bmatrix},
\]

\[
H_{x,5,5} = \begin{bmatrix}
0 & 0 & -1 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 1 \\
0 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0
\end{bmatrix},
\]

\[
H_{x,11,11} = \begin{bmatrix}
0 & 0 & -1 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 1 \\
0 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 1
\end{bmatrix},
\]

\[
H_{x,21,21} = \begin{bmatrix}
0 & 0 & -1 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 1 \\
0 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0
\end{bmatrix}.
\]

The eight filtering masks for vertical edge detection of eight directions are as follows:

\[
V_{y} = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 \\
0 & -1 & -1 & -1 & 0 \\
1 & 1 & 1 & 1 & 1 \\
0 & 1 & 1 & 1 & 0 \\
\end{bmatrix},
\]

\[
V_{y,5,5} = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 \\
0 & -1 & -1 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 1 \\
\end{bmatrix},
\]

\[
V_{y,11,11} = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 \\
0 & -1 & -1 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 1 \\
0 & 0 & 1 & 1 & 0 \\
\end{bmatrix},
\]

\[
V_{y,21,21} = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 \\
0 & -1 & -1 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 1 \\
0 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 1
\end{bmatrix}.
\]

For either diagonal, horizontal or vertical edge detection, we use element-by-element multiplication between the 5×5 image pattern in low-resolution image and the eight 5×5 filtering masks. After summing up the 25 products, it will produce eight filtering outputs, which will be normalized by the sum of the 25 corresponding filtering weights. The edge type of each 5×5 image pattern is determined as the one having the maximum “normalized” filtering output. However, if the difference between the maximum “normalized” filtering output and the minimum “normalized” filtering output of an image pattern is smaller than a threshold T, the image pattern is then classified as a small-edge one. We only perform the ADWS method, when the variance of its four neighbours (s) as shown in Fig. 3 of the unknown pixel is larger than a threshold T. The objective of testing the variance is to find out the edge region for performing the ADWS method. If the variance is smaller than T, bilinear interpolation is used instead of the ADWS in order to save computation. However, if the “normalized” filtering output difference is smaller than T, it means that this region does not have a major edge direction, therefore, we define the edge type of this region as the small edge type which means that this region contains some small edges with multiple directions.

### B. Adaptive Directional Elliptic Windows

According to the edge direction, let us define its elliptic window. If the edge type is small-edge, a circular window with radius 5 is used. The equation of the elliptic window is shown below:

\[
E_{\theta} = \frac{(x \cos(\theta) - y \sin(\theta))^2}{a} + \frac{(x \sin(\theta) - y \cos(\theta))^2}{b^2} = 1 \quad (8)
\]

where \( \theta = 0°, 22.5°, 45°, 67.5°, 90°, 112.5°, 135°, 157.5° \)

In order to get a sufficient number of the sample points for the Wiener filter, we choosing \( a = 3 \) and \( b = 7 \) in (8). Therefore, all the eight directional ellipses will be inside the 14×14 square window, which are the sample points (the gray region) in a particular direction as shown in Fig 4.

For either diagonal, horizontal or vertical edge detection, we use element-by-element multiplication between the 5×5 image pattern in low-resolution image and the eight 5×5 filtering masks. After summing up the 25 products, it will produce eight filtering outputs, which will be normalized by the sum of the 25 corresponding filtering weights. The edge type of each 5×5 image pattern is determined as the one having the maximum “normalized” filtering output. However, if the difference between the maximum “normalized” filtering output and the minimum “normalized” filtering output of an image pattern is smaller than a threshold T, the image pattern is then classified as a small-edge one. We only perform the ADWS method, when the variance of its four neighbours (s) as shown in Fig. 3 of the unknown pixel is larger than a threshold T. The objective of testing the variance is to find out the edge region for performing the ADWS method. If the variance is smaller than T, bilinear interpolation is used instead of the ADWS in order to save computation. However, if the “normalized” filtering output difference is smaller than T, it means that this region does not have a major edge direction, therefore, we define the edge type of this region as the small edge type which means that this region contains some small edges with multiple directions.

### B. Adaptive Directional Elliptic Windows

According to the edge direction, let us define its elliptic window. If the edge type is small-edge, a circular window with radius 5 is used. The equation of the elliptic window is shown below:

\[
E_{\theta} = \frac{(x \cos(\theta) - y \sin(\theta))^2}{a} + \frac{(x \sin(\theta) - y \cos(\theta))^2}{b^2} = 1 \quad (8)
\]

In order to get a sufficient number of the sample points for the Wiener filter, we choosing \( a = 3 \) and \( b = 7 \) in (8). Therefore, all the eight directional ellipses will be inside the 14×14 square window, which are the sample points (the gray region) in a particular direction as shown in Fig 4.

In order to reduce the covariance miss-match problem in the edge-directed interpolation method, we further propose to use multiple Directional Elliptic Windows according to the edge direction. Fig. 5 (a) shows a high-resolution block to be interpolated. As the direction of the edge (red line) is 45°, the 45° Elliptic Window is used. The Elliptic Window is arranged to slide along in the edge direction (45°) starting at the middle as shown in Fig. 5 (b), move up and right by two pixels as shown in Fig. 5(c) and move down and left by two pixels as shown in Fig. 5(d). Only the Elliptic Window that contains most edge information, i.e. Fig. 5 (d), will be chosen as the best candidate for this high-resolution block.
C. Window Selection

In the window selection process for the EDI, we select the elliptic window which has the lowest error energy. The cost function of the error energy can be derived from the N-order linear prediction equation.

Assume that we use an \( N \)-th order linear prediction equation to estimate the unknown pixel \( Y(k) \) which is denoted as \( Y'(k) \) by using the neighboring pixels, \( X(0), X(1), \ldots, X(N-1) \).

\[
Y'(k) = \sum_{n=0}^{N-1} \alpha_n X(n)
\]

In the matrix form, we can write (9) as below,

\[
Y'(k) = \alpha^T x
\]

Where vector \( \alpha = [\alpha_0, \alpha_1, \ldots, \alpha_{N-1}]^T \) and the vector \( x=[X(0), X(1), \ldots, X(N-1)]^T \).

Then, we can derive the equation of the cost function between the \( Y(k) \) and \( Y'(k) \) by using the Means Square Error (MSE).

\[
J_{\text{error}} = E[(Y(k) - Y'(k))^2]
\]

Substitute (2) and (10) into (11), we have

\[
J_{\text{error}} = E[(Y(k) - Y'(k))^2]
\]

\[
= E[(Y(k) - (R_{yy}^{-1} r_y)^T x)^2]
\]

\[
= E[(Y(k))^2] - 2(E[Y(k)]^T x)(R_{yy}^{-1} r_y)^T x +
\]

\[
+ (R_{yy}^{-1} r_y)^T x E[x^T x] R_{yy}^{-1} r_y
\]

\[
= E[(Y(k))^2] - 2(E[Y(k)]^T x)(R_{yy}^{-1} r_y)^T r_y +
\]

\[
+ (R_{yy}^{-1} r_y)^T R_{yy} (R_{yy}^{-1} r_y)
\]

\[
= E[(Y(k))^2] - 2(E[Y(k)]^T x)(R_{yy}^{-1} r_y)^T r_y + (R_{yy}^{-1} r_y)^T R_{yy} (R_{yy}^{-1} r_y)
\]

\[
= E[(Y(k))^2] - 2(E[Y(k)]^T x)(R_{yy}^{-1} r_y)^T r_y + (R_{yy}^{-1} r_y)^T R_{yy} (R_{yy}^{-1} r_y)
\]

\[
= E[(Y(k))^2] - 2(E[Y(k)]^T x)(R_{yy}^{-1} r_y)^T r_y + (R_{yy}^{-1} r_y)^T R_{yy} (R_{yy}^{-1} r_y)
\]

where \( E[Y(k)x] \) is the high-resolution cross-covariance \( r_y \) and \( E[xx^T] \) is the high-resolution auto-covariance \( R_{yy} \).

\( Y(k) \) and the high-resolution cross-covariance \( r_y \) are not available now, because \( Y(k) \) is to be predicted. To solve this problem, we use the geometric duality between the high-resolution covariance and low-resolution covariance. Therefore, the low-resolution covariance \( R_{yy}, r_y \) and \( y \) will be used instead, for the calculation. The final equation for the cost function is shown below.

\[
J_{\text{error}} = y^T y - r_y^T (R_{yy}^{-1} r_y)
\]

\[
= y^T y - r_y^T \alpha
\]

Using (12), we can find out the cost of each of the elliptic window, and then we can choose the elliptic window that has the lowest cost value which means it has the lowest error energy.

Note that (12) have been derived without considering the optimization using derivative that is usually used in the formulating the Wiener filter.

IV. EXPERIMENTAL RESULT

The proposed algorithm was implemented and compared with several conventional approaches in literature including bilinear interpolation, the NEDI method [12] and the MEDI method [13]. All the programs were write in C++ language and run on the same platform. The values of the thresholds \( T_s \) and \( T_e \) using in the proposed ADWS are 8 and 10 respectively. In the comparison, our proposed enhancement technique, ADWS, has been applied based on the MEDI method.

In order to compare the performance of these algorithms in terms of objective image quality, a low-resolution image was obtained by down sampling a high-resolution image by a factor of 2 and then we reconstructed it by conventional methods and our algorithm. We compare the interpolated high-resolution image with the original high-resolution image in terms of PSNR and visual quality. We have used various test images and the PSNR of all test images are summarized in Table 1.

In Table 1, when our proposed Adaptive Directional Window Selection method was applied to the MEDI, the PSNR is slightly increased.

Although, the proposed ADWS method does not have a large PSNR improvement, it results in a better objective quality than the NEDI and MEDI.

In Fig. 6, the interpolated image (d) using bilinear interpolation contains many aliasing artifacts so it has the worst visual quality. The interpolated image (e) using NEDI resulted in a more continuous edge image, but it is still affected by the ringing effect which shows inside of the red ellipse. The interpolated image (f) using MEDI contain a lot of ringing effects as shown inside of the red ellipse. We can see that the interpolated image (g) using our proposed method ADWS in MEDI can reduce most of the ringing effect in the MEDI and NEDI methods.

Moreover, in Fig. 7, the interpolated image (d) using the bilinear interpolation cannot perform well in the texture areas. The interpolated image (e) using the NEDI resulted in a more accurate edge prediction than the bilinear interpolation, but it is still affected by the ringing effect as shown inside of the red ellipses. This also cause the discontinue of the edge region. The interpolated image (f) using MEDI contains a lot of ringing effects as shown inside of the red ellipses. We can see that the interpolated image (g) using our proposed method ADWS in...
MEDI can reduce most of the ringing effect in the MEDI and NEDI method and keep the continues of the edge region.

Figs. 6 (g) and 7 (g) also show that our proposed method (ADWS) can produce an interpolated image which have sharper and more continuous edges than other interpolation methods (see blue ellipses in Figs. 6 and 7). It is because using the edge direction information together with the Adaptive directional elliptic window can reflect the low-resolution statistic more accurately than the normal non-directional square window in MEDI and NEDI. Although the MEDI method makes use of the multi-square windows to fit the edge orientation, the selection of the square windows in MEDI is not accurate enough. This cannot select the best window which contains most of the edge information. Therefore, in some texture regions, the performance of MEDI is worse than that of NEDI.

Another advantage of the proposed method (ADWS) is the edge performance. Areas near to the end of the edge or near to the texture region with multi-direction edges are improving. In Fig. 8, there are many ringing effects that are near to the main edge, as shown inside of the red ellipses. It is because the square window is non-directional and it will always gives wrong statistic information, so that the Wiener filter cannot accurately predict the true direction of the edge, so ringing effect happens. In Fig. 8, we can see that the direction of the ringing effect inside of the red ellipses is about 135° which is the direction of the metal rod in the center of the red ellipses, but the true direction of the edge is about 45°. This mismatch problem is caused by the inaccuracy of statistic information in the square window using in the NEDI and MEDI. After applying the proposed ADWS in MEDI, the ringing effect is removed, which gives the best visual quality of the interpolated image as shown in Fig. 8 e.
In this paper, we have proposed an adaptive directional window selection (ADWS) for the EDI. It makes use of the edge information to find the practical directional elliptic window which is suitable for edges. We arrange the elliptic window to slide along on the edge direction and then we select the best elliptic window as the sample points for the optimization in the Wiener filter. Experimental results show that the proposed ADWS can reduce most of the ringing effect and artifact in the interpolated image compared to that of the NEDI and MEDI. It can overcome the existing problem of the NEDI in high frequency content or texture region by considering multiple directional elliptic windows. Moreover, the window selection method in ADWS is accurate, so that the prediction error in the interpolated image is reduced compared to the NEDI and MEDI. As a result, the proposed algorithm produces interpolated images with better objective and subjective qualities compared to that of the conventional interpolation methods.

Table 1: The PSNR of Interpolated Color (RGB) Test Images by Different Interpolation Methods.

<table>
<thead>
<tr>
<th>Image</th>
<th>Bilinear</th>
<th>NEDI</th>
<th>MEDI</th>
<th>MEDI (ADWS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>kodim01</td>
<td>23.75</td>
<td>23.60</td>
<td>23.60</td>
<td>23.60</td>
</tr>
<tr>
<td>kodim02</td>
<td>30.70</td>
<td>30.64</td>
<td>30.65</td>
<td>30.65</td>
</tr>
<tr>
<td>kodim03</td>
<td>31.42</td>
<td>31.61</td>
<td>31.61</td>
<td>31.61</td>
</tr>
<tr>
<td>kodim04</td>
<td>30.48</td>
<td>30.35</td>
<td>30.37</td>
<td>30.37</td>
</tr>
<tr>
<td>kodim05</td>
<td>23.78</td>
<td>23.94</td>
<td>23.94</td>
<td>23.94</td>
</tr>
<tr>
<td>kodim08</td>
<td>21.09</td>
<td>20.95</td>
<td>20.95</td>
<td>20.95</td>
</tr>
<tr>
<td>kodim09</td>
<td>29.00</td>
<td>29.04</td>
<td>29.05</td>
<td>29.05</td>
</tr>
<tr>
<td>kodim10</td>
<td>28.96</td>
<td>28.92</td>
<td>28.92</td>
<td>28.92</td>
</tr>
<tr>
<td>kodim11</td>
<td>26.82</td>
<td>26.78</td>
<td>26.78</td>
<td>26.78</td>
</tr>
<tr>
<td>kodim13</td>
<td>26.46</td>
<td>26.43</td>
<td>26.43</td>
<td>26.43</td>
</tr>
<tr>
<td>kodim16</td>
<td>29.16</td>
<td>29.03</td>
<td>29.02</td>
<td>29.02</td>
</tr>
<tr>
<td>kodim17</td>
<td>29.50</td>
<td>29.47</td>
<td>29.48</td>
<td>29.48</td>
</tr>
<tr>
<td>kodim18</td>
<td>25.72</td>
<td>25.64</td>
<td>25.64</td>
<td>25.64</td>
</tr>
<tr>
<td>kodim21</td>
<td>26.05</td>
<td>25.89</td>
<td>25.89</td>
<td>25.89</td>
</tr>
<tr>
<td>kodim23</td>
<td>31.26</td>
<td>31.37</td>
<td>31.37</td>
<td>31.37</td>
</tr>
</tbody>
</table>

| Average | 27.34 | 27.35 | 27.30 | 27.31 |

V. CONCLUSION

REFERENCES


