Finite Element Based Interpolation Methods for Spatial and Temporal Resolution Enhancement for Image Sequences

Yan Wu

A Thesis

in

The Department

of

Electrical and Computer Engineering

Presented in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy at
Concordia University
Montreal, Quebec, Canada

March 2010

Your file Votre référence
ISBN: 978-0-494-67372-0
Our file Notre référence
ISBN: 978-0-494-67372-0

ABSTRACT

Finite Element Based Interpolation Methods for Spatial and Temporal Resolution Enhancement for Image Sequences

Yan Wu, Ph. D.

Concordia University, 2010

Spatial resolution enhancement is a process for reconstructing a high resolution image from a low resolution image, whereas temporal resolution enhancement of encoded video aims at interpolating the skipped frames, making use of two successively received frames. In this thesis, a new image interpolation model, called the *generalized image interpolation model*, is developed in order to devise new techniques for spatial resolution enhancement of images, and temporal resolution enhancement of encoded video sequences. The interpolation model is based on the finite element method, and takes into account the unknown neighboring pixels, and therefore is capable of interpolating a collection of unknown pixels with an arbitrary shape, while providing a spatial continuity between the unknown pixels.

Based on the generalized interpolation model, an edge-preserving iterative refinement scheme for spatial resolution enhancement of images is proposed. This scheme exploits not only the neighboring pixels whose values are known, but also takes into account those with unknown values. It is shown that the edge-preserving iterative refinement process maintains the smooth variation along a dominant edge in the up-

scaled image. Simulation results show that the proposed scheme results in up-scaled images with subjective and objective qualities, which are better than those of the existing interpolation schemes. Further, the scheme is also shown to be capable of up-scaling an image by an arbitrary magnification factor, without resorting to extra steps, or the use of any conventional interpolation method.

Next, error concealment-based MCI schemes are also presented for temporal resolution enhancement of encoded video sequences. These schemes are also based on the generalized image interpolation model, and need no pixel classification, thus reducing substantially the computational complexity. They are shown to be capable of concealing the errors in the homogeneous regions as well as in regions containing sharp edges. Experiments are carried out showing that the proposed schemes result in reconstructed frames having a better visual quality and a lower computational complexity than that provided by the existing techniques.

To my loving family

ACKNOWLEDGEMENTS

I am heartily thankful to my supervisors, Dr. M. Omair Ahmad and Dr. M.N.S. Swamy, for their guidance, support and continuous encouragement throughout the course of doctoral study. Their invaluable advices and feedback have guided me though some difficult period of times. It would not have been possible to complete this thesis but for their patience, kindness and support.

I also want to thank my colleagues at Miranda Technologies. I appreciate the friendship and encouragement from Mr. Michel Proulx, Mr. Garnet Carter, Mr. Keith Delpeche, and Mrs. Jean-Marie Edades.

I am deeply indebted to my parents for their love and encouragement. My profound gratitude goes to my wife Wei Wang, for her patience, understanding, and unconditional support and help.

Contents

List of Figures

List of Tables

List of Abbreviations

CIF	Common Intermediate Format
DB-FMCI	Deformable Block-based Frame Motion-Compensated Interpolation
EPIR	Edge-Preserving Iterative Refinement
EDIM	Edge-Directed Interpolation Method
ERIR	Edge-Reuniting Iterative Refinement
FR	Frame Repetition
FA	Frame Averaging
FEM	Finite Element Method
FMCI	Fast Motion-Compensated Interpolation
GIR	Generic Iterative Refinement
I-MCI	Inertia Motion-Compensated Interpolation
MCI	Motion-Compensated Interpolation
POCS	Projection Onto Convex Sets
PSNR	Peak Signal to Noise Ratio
QCIF	Quarter Common Intermediate Format
ROI	Region of Interest
SMVF	Scaled Motion Vector Field
VQ	Vector Quantization

List of Symbols

$a(u(x,y),v(x,y))$	Variational function
A	An unknown pixel
B^e	Coefficient matrix
B_i	Image block in a frame
BLK	Sub-image block
BLK^u	Sub-image block of up-scaled image $H(m,n)$
C	Boundary for heat conduction region R
$C_V(i,j)$	Vertical interpolation function for an up-scaled image
$C_H(i,j)$	Horizontal interpolation function for an up-scaled image
D	Domain for a low resolution image
D^u	Domain for an up-scaled image $H(m,n)$
e	Triangular elements in the domain for heat conduction problem
e_v	Triangular elements in an image
FRM_t	Received image at time t
$\hat{FRM}_{t-\alpha}$	Interpolated frame at time $t-\alpha$
$F^c_{t-\alpha}(x,y)$	Continuous function for interpolated frame $\hat{F}_{t-\alpha}$
$G(i,j)$	The magnitude of the gradient at (i,j)

$H(m,n)$	An up-scaled high resolution image
$H(x,y)$	An continuous luminous function
H	Height of an image
I	Globe node position for a triangular element
I_{mg}	An image
J	Globe node position for a triangular element
k^e	Local coefficient matrix for a triangular element
K	Globe node position for a triangular element
K^e	Global coefficient matrix for a triangular element
$L(i,j)$	Low resolution image or image with unknown pixels
$L(x,y)$	A continuous luminance function
m_G	The arithmetic mean of the gradient values of all the pixels in $L(i,j)$
$\vec{n}$	The unit vector normal to the boundary C and directed out of the region R
$N_i^e(x,y)$	Shape function for a triangular function
$\vec{p}$	The position vector of a pixel at point P in a frame
P	Pixel position in a frame
Q	Number of non-overlapping blocks in a compensated frame
R	The region for a heat conduction problem
R_r	Register associated with different orientations

[illegible]	Register with the largest value
[illegible]	The area of an triangular element
$S_V(i,j)$	Vertical interpolation function for iterative refinement procedure
$S_H(i,j)$	Horizontal interpolation function for iterative refinement procedure
$T(x,y)$	The temperature function
Th	Threshold for the magnitude of gradient
u	Magnification factor
U	The set of unknown nodes
$\vec{V}_i$	Motion vector for an image block
W	Width of an image
α	A temporal position in the interval from 0 to 1
$\beta_V(i,j)$	Vertical interpolation function used for an interpolated frame
$\beta_H(i,j)$	Horizontal interpolation function used for an interpolated frame
$\theta(i,j)$	The angular direction of the gradient at (i,j)
K	The set of known nodes
Λ	Domain for continuous function $F^c_{t-\alpha}(x,y)$
Λ^e	Incidence matrix
Γ	The set containing all the nodes in domain Λ
$\vartheta(x,y)$	Shape function for a triangular element in an interpolated frame

Φ	The domain for function $H(x, y)$
ρ	Coefficient for calculation of edge detection threshold
φ	Number of non-overlapping triangular elements in domain Φ
$\varphi(x, y)$	Shape function for a triangular element in an image with unknown pixels
Ψ	Coefficient matrix for the global nodes of the elements
Ω	A domain for a continuous luminance function
$\omega(x, y)$	Shape function for a triangular element in an up-scaled image

Chapter 1

Introduction

1.1 General

Digital video consists of a sequence of digital images. Spatial resolution of the images along with the temporal resolution of the image sequence, are amongst the most important attributes of a digital video. Spatial resolution of a digital image is decided by the number of pixels utilized in the construction of the image. An image with a higher spatial resolution is composed of more pixels than those of a lower spatial resolution. Temporal resolution of a digital video refers to the frequency at which the frames are captured, recorded or displayed. The higher the frequency, the better or the finer the temporal resolution is said to be. Although ideally one would like to have a digital video with a high spatial resolution as well as a high temporal resolution, it is often not the case in real life applications, due to various reasons. For example, due to physical limitations and/or cost effectiveness of image sensors, it may not be possible to produce a digital image with the desired spatial resolution. Also, due to the limitations of the bandwidth and/or storage space, a digital video encoder may not encode all the frames of a video. Therefore, to achieve a higher compression ratio, frame dropping or frame skipping is

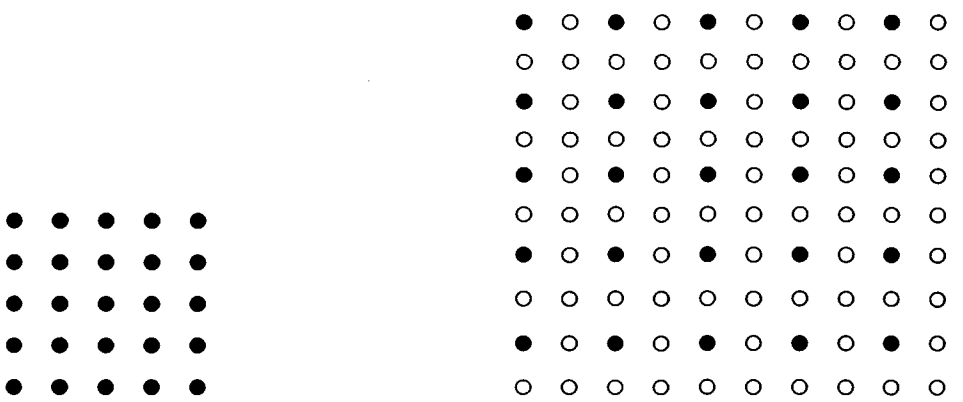
commonly used in the existing video coding techniques. Consequently, the decoded video with dropped frames will result in jerky and visually not so pleasant frames.

Spatial resolution enhancement consists of reconstructing a high resolution image from a low resolution image, whereas temporal resolution enhancement of encoded video aims at interpolating the skipped frames, making use of two successively received frames, so that the decoded video can be played back at the normal frame rate.

1.2 Spatial Resolution Enhancement of Images

In broadcast industry, spatial resolution enhancement is an essential step for up-converting a standard definition video to a high definition video, and plays an important role in the present day television station's infrastructure equipment. In the printing industry, spatial resolution enhancement technique is applied to produce high quality prints for posters, magazines, and commercial catalogs. This technique is also widely used in other fields, such as digital photography, medical imaging, remote sensing, and surveillance. The basic idea of spatial resolution enhancement is illustrated in Fig. 1.1, in which a low resolution image is up-scaled to a high resolution image using spatial resolution enhancement with a magnification of two.

Conventional spatial resolution enhancement techniques include nearest-neighbor method, bilinear interpolation method, and various cubic interpolation methods [1]-[3], which have the advantage of low computational complexity. These interpolation methods are based on the scene-independent model, which cannot spatially adapt the interpolation coefficients to match fast changing pixel structures in a certain area of an image, such as



(a) (b)

Figure 1.1: (a) A low resolution image (pixels represented in black dots). (b) The up-scaled image from its low resolution version (white dots representing the pixels to be interpolated)

an area around the edges. Therefore, in the up-scaled image, these interpolation techniques generate some noticeable artifacts, such as block effects, blurred details, and jagged edges. Hence, enhancement of image resolution through interpolation, while at the same time reducing these artifacts, is an important topic of interest in image processing. Many new interpolation schemes [4]-[21] have been proposed to improve the subjective quality of the interpolated images. Some schemes use the projection onto convex sets (POCS) model to constrain the edge continuity and then find the appropriate estimation for the interpolated pixels [4]-[6]. In wavelet-based interpolation schemes, the similarity between different scales of a decomposed image is exploited in a way that the high resolution details are estimated from the low resolution data [7]-[9]. Some schemes, based on the vector quantization (VQ) and morphological filtering techniques, have also been proposed to enhance the spatial resolution of images [10], [11]. Many of the recently proposed spatial resolution enhancement schemes are edge-directed, and emphasize on the performance in the edge area of an up-scaled image [12]-[21]. In general, an edge-directed interpolation scheme imposes a scene-dependent model,

focusing on the local structure of a detected edge. As a result, such a scheme is capable of adapting the interpolation coefficients so that the interpolated pixels better fit the structure in the detected edge areas.

1.3 Temporal Resolution Enhancement of Encoded Video Sequences

In multimedia applications, such as video telephony, video conferencing and video streaming, a video encoder needs to cope with varying transmission bandwidth. In order to meet the constraint due to the limitations of the bandwidth, it is a common practice to sacrifice the temporal resolution, and use frame dropping or frame skipping to reduce the date rate of the encoded video. To playback the encoded video at the normal frame rate, the skipped frames have to be interpolated making use of two successively received frames, since otherwise, the playback of the decoded video with dropped frames will result in motion jerkiness. The basic idea of the temporal resolution enhancement is illustrated in Fig. 1.2, in which one out of every two frames in the source video is dropped at the transmitter and interpolated at the receiver.

Temporal resolution enhancement of encoded video sequences has drawn a great deal of attention in the development of efficient schemes. Early attempts in frame interpolation included frame repetition (FR) and frame averaging (FA) [22]. In an FR scheme, the system simply duplicates the previous frame before the next one arrives. In the case of an FA scheme, the average pixel values of the previous and current frames are taken into consideration for reconstructing the interpolated frame. These FR and FA

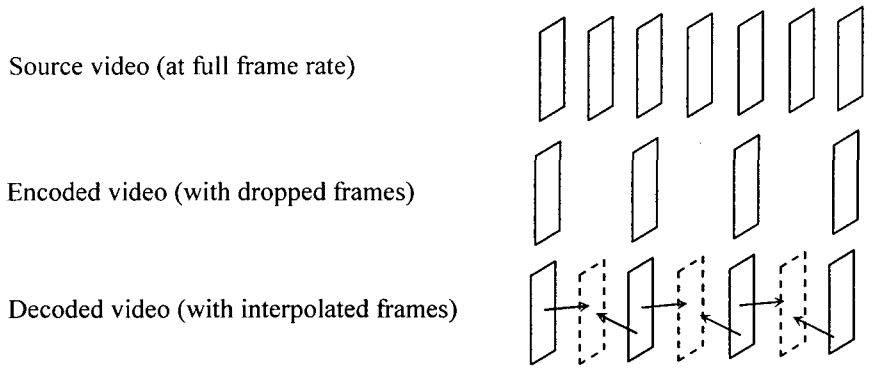


Figure 1.2: Basic idea of temporal resolution enhancement for encoded video

techniques are rather straightforward; but, for video sequences containing fast motions, they will respectively result in jerky and blurred images. Since the motion between the frames is the major cause of these problems, the authors in [22] have used the motion information between the transmitted frames to perform the frame interpolation. Since then, many motion-compensated interpolation (MCI) schemes have been developed [23]-[48].

Essentially, there are two types of MCI schemes, pixel-based and block-based. In the pixel-based schemes [22]-[35], a recursive pixel motion estimation is generally required, and the classification of the pixels is based on the moving objects and the static, covered and uncovered backgrounds. Obviously, this type of scheme is computationally very demanding, and useful only in sophisticated applications such as display-frequency conversion, film-to-tape conversion, and video standard conversion. The block-based MCI schemes [36]-[48] exploit the block motion vectors decoded at the receiver, so that the computationally intensive pixel motion estimation can be avoided. Therefore, for the purpose of temporal resolution enhancement of encoded video sequences, this scheme can be used in the conventional video applications, which employ block-based, motion-compensated solutions such as H.26X, or MPEG standards.

1.4 Motivation and Objectives

The challenge of any spatial resolution enhancement for an image is to enhance the resolution through interpolation, without introducing artifacts, such as block effects, blurred details, and jagged edges, while at the same time keeping the computational complexity as low as possible.

Although quite a few edge-directed interpolation schemes have been developed, they do not take into account the neighboring unknown pixels when interpolating a pixel for an up-scaled image. In the existing interpolation schemes, an unknown pixel is interpolated based on the estimation made only with the neighboring known pixels. In a natural image, the image textures and edges consist of pixels that are related to one another. Hence, including only the neighboring known pixels in the interpolation process does not give a very natural looking reconstructed image, especially in areas with fine textures.

All the existing edge-directed interpolation schemes can only be applied when the magnification factor is an integer power of two, since each of these schemes is attached to a specific interpolation model, which is commonly designed for a magnification factor of two. Consequently, these schemes need the use of some conventional interpolation methods when magnification factor is not a power of two. Obviously, the advantages associated with their schemes cannot be fully realized in such cases in view of the need to use conventional interpolation methods as well.

Since the existing interpolation schemes interpolate unknown pixels based only on the estimation made with known pixels, these schemes require more than one step to complete the interpolation process even when an image is up-scaled by a magnification

factor of two. Obviously, such a scheme needs to be applied n times to up-scale an image by a magnification factor of 2^n (n an integer greater than one). Thus, the question as to how to complete the interpolation process without multiple interpolation steps still remains unanswered.

Motivated by the challenges and aiming at solving the above mentioned problems, this thesis first focuses on developing a more general interpolation model that takes into account not only the neighboring known pixels, but also the neighboring pixels with unknown values in order to provide a spatial continuity between the unknown pixels as well. Using this model, a new technique for spatial resolution enhancement is developed. The technique should be general enough so that it is capable of up-scaling an image by an arbitrary integer magnification factor and at the same time it needs to be applied only once irrespective of the value of the magnification factor.

As mentioned earlier, the block-based MCI scheme is preferable to the pixel-based MCI scheme for the temporal resolution enhancement of encoded video sequences. However, there are two major problems related to the block-based MCI scheme, namely, as to how to deal with overlapped pixels in the interpolated frames, and as to how to handle the holes left in the interpolated frames. In existing methods, an averaging technique is often used to handle the overlapped pixels, while simple techniques such as the pixel averaging or repetition [43], [44], are used to fill the holes. These simple techniques result in interpolated frames that suffer from blur and stripe effects. In view of this, we intend to develop in this thesis a block-based MCI scheme that can conceal the interpolation errors caused by the overlapped pixels and holes at a relatively low computational complexity.

From the above discussion, it is clear that for spatial resolution enhancement as well as temporal resolution enhancement, there is a need to develop a new image interpolation technique to meet the challenges mentioned above. Finite element method (FEM) is a powerful numerical mathematics tool, which offers the simplicity of piecewise approximation of a function given its values at discrete points. It has been employed for obtaining the numerical solution to a wide variety of engineering problems, such as those in the field of mechanical engineering, aeronautical, biomechanical, and automotive industries. In view of this, the FEM is chosen in this thesis to develop an image interpolation model, and this model is utilized to design a new interpolation algorithm for the enhancement of spatial and temporal resolution.

1.5 Organizations of the Thesis

This thesis is organized as follows. In Chapter 2, a review of the background material concerning the spatial resolution enhancement for images, focusing on the state of the art edge-directed interpolation schemes, is presented. Then, a review on block-based MCI schemes for temporal resolution enhancement of encoded video sequences is given. In addition, since the FEM is chosen for the development of a new image interpolation technique, basic concepts concerning FEM are also presented. In Chapter 3, the development of a generalized image interpolation approach based on FEM method is presented [47], [48]. In Chapter 4, an iterative refinement interpolation scheme based on the generalized image interpolation approach is developed [49], in order to obtain an up-scaled image from a low resolution image. In Chapter 5, an error concealment-based MCI scheme for temporal resolution enhancement of encoded video sequence is developed

[48], [50]. Finally, in Chapter 6, the main contributions of the thesis are highlighted, and some possible future research directions suggested.

Chapter 2

Background and Literature Review

In this chapter, some basic concepts related to edge-directed spatial resolution enhancement, as well as block-based temporal resolution enhancement, are introduced. Relevant literature is reviewed, and some state of the art techniques discussed. In addition, fundamentals of finite element methods are introduced in this chapter, since it will be used in designing a new image interpolation model.

2.1 Edge-directed Spatial Resolution Enhancement of Images

2.1.1 Fundamentals for spatial resolution enhancement

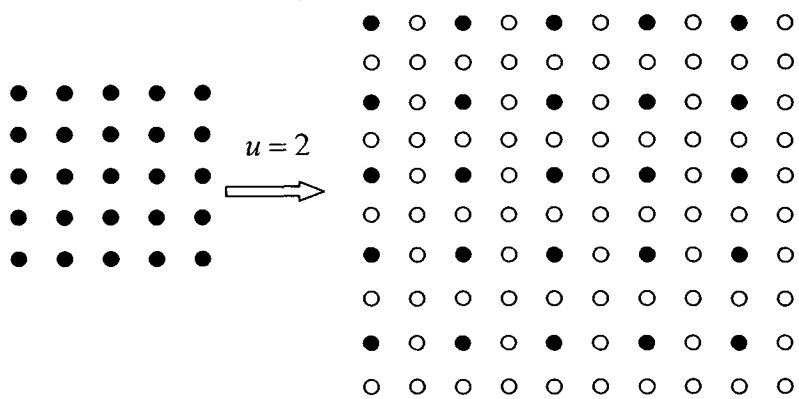
Spatial resolution enhancement can be considered as the process of obtaining a high resolution image by up-scaling a low resolution image. This process can be formulated as follows. Consider a low resolution image $L(i,j)$ of size $(W \times H)$, $(i,j) \in D$, where D is given by

$$D = (0,1,\ldots,W-1) \times (0,1,\ldots,H-1) \tag{2.1}$$

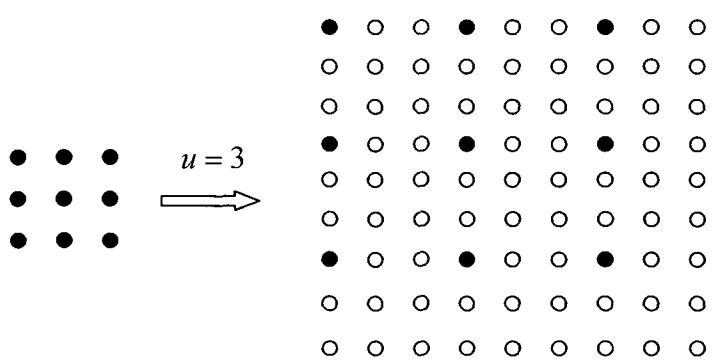
To scale $L(i,j)$ by a magnification factor u ($u > 1$), let us form an image $H(m,n)$, $(m,n) \in D^u$, where

$$D^u = (0,1,\ldots,uW-1)\times(0,1,\ldots,uH-1) \tag{2.2}$$

Therefore, at $(ui,\ uj)$, $H(m,n)$ has known pixels values, which are equal to $L(i,j)$. The task is to interpolate the unknown pixels at (m,n), $(m \neq ui \text{ or } n \neq uj)$, based on the known pixels at $(ui,\ uj)$. In Fig. 2.1, we illustrate two simple examples of the image up-scaling process for $u = 2$ and $u = 3$.



(a)



(b)

Figure 2.1: (a) An image of size 5×5 is up-scaled to an image of size 10×10, with $u = 2$. (b) An image of size 3×3 is up-scaled to an image of size 9×9 with $u = 3$. Black dots represent pixels from low resolution images, white dots representing the pixels to be interpolated

As discussed in the previous chapter, linear interpolation methods are commonly used for spatial resolution enhancement of images, but are error prone in regions with edges or fine textures. Although statistically these are errors with a small population compared to the size of an up-scaled image, their unpleasant effects (such as block effects, blurred details, and jagged edges) on the visual quality of the up-scaled image could be quite large. This is so due to the fact that the edges are one of the most prevalent features of images in human visual system. This also explains as to why most of the recent spatial resolution enhancement schemes tend to focus on the performance in the edge area of the up-scaled image.

2.1.2 Edge-directed spatial resolution enhancement techniques

Many researchers have adopted the approach of edge-guided interpolation in their techniques for spatial resolution enhancement. Some of these techniques [12]-[15] employ directional interpolation that requires an accurate prediction of the edge orientation in the up-scaled high resolution image. In [16], the authors have proposed a scheme that employs edge detection and predefined templates, in order to improve the visual quality of the up-scaled images. They use a certain number of edge patterns in the image interpolator, the parameters of which are adapted based on the detected edges. Zhang and Wu [17] have proposed a scheme to interpolate unknown pixels in multiple directions, and then apply data fusion to the interpolated results. It is to be noted that the prediction of the presence of very thin and well-linked edges in the high resolution image is computationally demanding. Moreover, in these approaches, edge orientation is quantized into a number of discrete levels, and this affects the accuracy of the

interpolation and can cause exaggeration of the edges, since an edge existing in a natural image is arbitrarily oriented.

Based on the assumption that the magnification factor is two, Lin and Orchard [18] have proposed a method to estimate the covariance of the up-scaled image from that of its low-resolution image and then to interpolate the unknown pixels based on these results. This scheme has been considered as one of the best amongst the edge-directed interpolation methods. However, the computational complexity can be as high as 1300 multiplications per pixel. Therefore, the authors in [18] have employed a hybrid approach, wherein an activity measure is first used to determine if a pixel is an edge pixel or not, and then depending on whether the pixel is an edge pixel or not, different interpolation methods are employed.

Wu and Zhang [19] have proposed a method wherein a global texture orientation map is first generated, and then refined by a kernel Fisher discriminant. However, a training set needs to be employed to gather prior knowledge of the orientation map before the refinement is carried out. This requirement may not be met under all conditions. Wang and Ward [20] have proposed an edge-directed image expansion scheme, wherein the edge pixels are found in a low resolution image, and then mapped on to the up-scaled image. On the up-scaled image, the pixels corresponding to the edge pixels in the low resolution image are assigned the average value of all these pixels. Although, as a part of the post-processing, edge sharpening is performed, the assignment of the same value to several pixels introduces a blur effect in the edge area. Zhang and Wu [21] have developed an image interpolation scheme, which is based on adaptive 2-D autoregressive modeling and soft-decision estimation. This approach is capable of preserving spatial

coherence of the interpolated images, and therefore, presents a good visual quality for the up-scaled image. However, the interpolation model is set up specifically for the case when the magnification factor is equal to two and hence, this technique cannot be applied for an arbitrary magnification factor. In view of this, the authors in [21] have applied bicubic interpolation in combination with their proposed scheme to realize the up-scaling of an image for a magnification factor of three.

It is to be noted that these existing edge-directed interpolation schemes do not take into account the neighboring unknown pixels when interpolating a pixel for an up-scaled image. In these schemes, an unknown pixel is interpolated based on the estimation made only with the neighboring known pixels. However, in a natural image, the image textures and edges consist of pixels that are related to one another. Hence, including only the neighboring known pixels in the interpolation process does not give a very satisfactory reconstructed image, especially in areas with fine textures. Moreover, these schemes require more than one step to complete the interpolation process even when an image is up-scaled by a magnification factor of two. When up-scaling an image by a magnification factor of 2^n (n an integer greater than one), such a scheme needs to be applied n times. Another issue with the existing edge-directed interpolation schemes is that these have been developed based on the assumption that the magnification factor is two. Therefore, these methods have difficulty in handling the case when the magnification factor is not an integer power of two. A common practice to go around this problem has been to resort to conventional interpolation methods, such as bilinear or bicubic interpolation methods as an additional step after up-scaling the image by a power of two using one of the methods. Obviously, in such a scheme, the advantages associated

with their schemes cannot be fully realized in view of the need to use conventional interpolation methods as well.

From above discussion, it is seen that there is a need for developing an edge-directed interpolation scheme for temporal resolution enhancement, which needs to be applied only once to interpolate all the unknown pixels in the up-scaled image, irrespective of the value of the magnification factor. In order to meet this challenge, one has to find an appropriate mathematical approach to design a generalized interpolation model, and derive a formula for interpolating all the unknown pixels, independent of the value of u.

2.2 Block-based MCI Schemes for Temporal Resolution Enhancement of Encoded Video Sequences

In this section, block-based MCI schemes are first introduced, and then the reasons for the occurrence of interpolation errors in such schemes analyzed. Some state of the art techniques relating to this topic are reviewed. The difficulty of concealing the interpolation errors without employing pixel classification is discussed. The feasibility of using a spatial interpolation method to deal with the interpolation errors is also explored.

2.2.1 Fundamentals for block-based MCI schemes

A block-based MCI scheme for temporal resolution enhancement exploits the block motion vectors available at a decoder, thus avoiding the need for the computationally demanding motion estimation. Let FRM_{t-1} and FRM_t be successive reconstructed frames at the receiving end at times $t-1$ and t, respectively. Let $\vec{p}$ denote the position

vector of a pixel at point P in a frame, $F_t(\vec{p})$ represent the pixel intensity at P inside the frame FRM_t, and $\vec{V}_i$ refer to the transmitted motion vector of a block B_i in the frame FRM_t. Obviously, $\vec{V}_i$ is assigned to any pixel at $P \in B_i$. Let $\hat{FRM}_{t-\alpha}$ be the interpolated frame at time $t-\alpha$. Considering a constant-speed motion model, the motion vector between FRM_t and $\hat{FRM}_{t-\alpha}$ is given by

$$\vec{V}_i^{\,\alpha} = \alpha \cdot \vec{V}_i \quad 0 < \alpha < 1 . \tag{2.3}$$

The relationship between these two vectors $\vec{V}_i$ and $\vec{V}_i^{\,\alpha}$ is shown in Fig. 2.2. For any $P \in B_i$, the corresponding location P_α in $\hat{FRM}_{t-\alpha}$ is found by using

$$\vec{p}_\alpha = \vec{p} + \left[\vec{V}_i^{\,\alpha}\right]_{round} \tag{2.4}$$

where $[\,\cdot\,]_{round}$ represents the rounding operation, which rounds the motion vector to the nearest integer within the image dimension. Then, the pixel value at P_α is given by

$$\hat{F}_{t-\alpha}(\vec{p}_\alpha) = F_t(\vec{p}) \tag{2.5}$$

which we refer to as the backward motion compensation that yields the interpolated

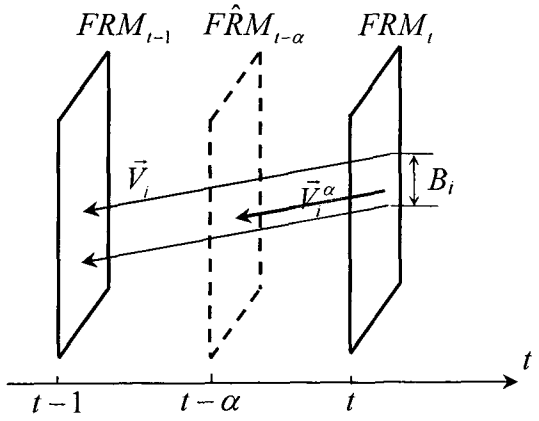


Figure 2.2: Relationship between the motion vector $\vec{V}_i$ and $\vec{V}_i^{\,\alpha}$

frame. This motion compensation process denoted by (2.5) may cause the occurrence of overlapped as well as unfilled pixels, and is a result of the mechanism used for the block-based motion estimation and motion compensation prior to the interpolation. Since for each non-overlapped motion block B_i in the frame FRM_t, the optimal motion vector $\vec{V}_i$ that points to its matching block in the frame FRM_{t-1} is found based on some measure of distortion, and transmitted, some of these matching blocks could be overlapped in the frame FRM_{t-1}. If we consider the case of the halfway position of a motion vector in the compensated frame $F\hat{R}M_{t-\alpha}$ ($\alpha = 0.5$), some of the predicted blocks B_i' ($i = 1,2,3,4$) could also be overlapped. In Fig. 2.3 (a), the motions of all the pixels within one motion block are represented by one motion vector, which denotes the motion of B_i from frame FRM_t to FRM_{t-1}. It is normal that different motion blocks have different motion vectors, which may make the motions of the pixels along the boundary of two adjacent motion blocks not to be consistent. In Fig. 2.3(b), after applying the constant motion model, we have the derived motion vector $\vec{V}_i^{\alpha}$ of the motion block B_i, which represents the motion of B_i from the frame FRM_t to the frame $F\hat{R}M_{t-\alpha}$. In $F\hat{R}M_{t-\alpha}$, the predicted block, B_i' (the shaded blocks in Fig. 2.3(c)), can be obtained. Therefore, some regions in the compensated frames could experience the problem of overlap or the unavailability of the pixel values. That is, some pixels in the compensated frame may be interpolated multiple times, or the pixel values at some of the pixel positions may not be available, leaving some "*holes*" between the blocks. In Fig. 2.3(c), we can see the unfilled region between B_1' and B_2', as well as that between the B_3' and B_4'. The values of the overlapped

and unfilled pixels cannot be predicted correctly, giving rise to errors that we

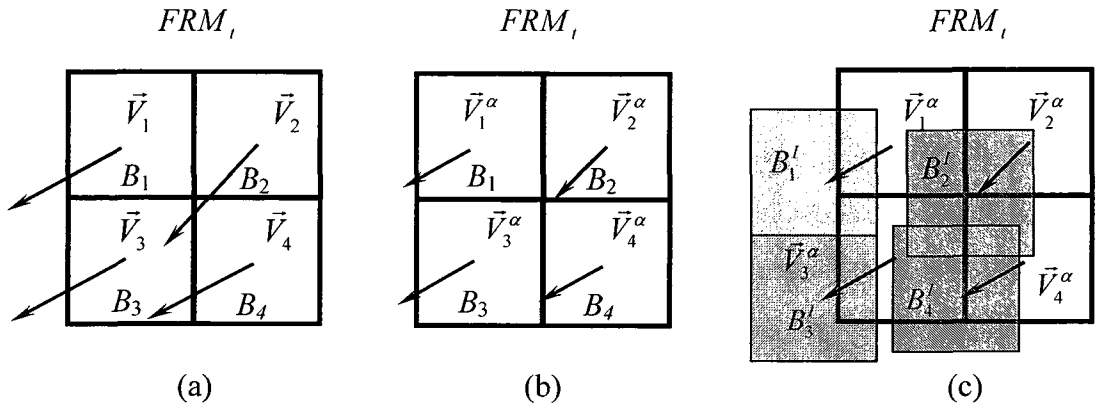


Figure 2.3: Illustration of the cause to the interpolation errors.

refer to as the interpolation errors for the interpolated frames. Usually, such interpolation errors occur in regions that are sensitive to the human eye within a frame, such as the area containing the head and shoulders, where the motion is likely to be relatively high and non-uniform. We shall refer to this area as the region of interest (ROI). If the interpolation errors are not concealed properly, the interpolated frame has an annoying stripe effect in the ROI, leading to an unsatisfactory subjective quality.

2.2.2 Block-based MCI schemes

Recent research in block-based MCI schemes for temporal resolution enhancement of encoded video sequences can be classified into two categories: MCI schemes based on proprietary video codec, and MCI schemes based on conventional video compression standards such as H.26X, or MPEG-1 /2/4. In the first category, the schemes are based on the proprietary codec, which supports special features in the encoder, so that extra information is obtained for the use by the MCI scheme at the decoder. The scheme developed in [36] considers multiple motion vectors for a single block to achieve a better

quality for the interpolated frames. This scheme requires extra motion information obtained from the video encoder. In [37], the MCI is performed using the object-based interpretation of the video at the encoder, so that the motion and segmentation information are available at the decoder for temporal resolution enhancement; obviously, it is based on a proprietary codec. In [38], a block-based fast motion-compensated interpolation (FMCI) scheme is proposed. In this scheme, there is an add-on module at the encoder to adaptively choose a variable number of skipped frames, so that the video coding with a variable frame rate is achieved. Due to the use of the add-on module on top of an existing video encoder, the standard bitstream syntax must be modified to accommodate this. Similarly, in [39], a MCI module is embedded in the encoder loop in order to improve the frame interpolation accuracy at the receiver. All these schemes are based on the proprietary encoders, which may not be acceptable for many video communication systems that are compliant to conventional video compression standards.

In the second category, the block-based MCI schemes are based on conventional video encoding standards, and work solely with the decoded video stream; they do not require any extra information to be obtained from the encoder. Therefore, they can be applied to most of the existing video communication systems. The basic idea of such a scheme is illustrated in Fig. 2.4. As seen from this figure, the motion information is part of the encoded bitstream from a standard video encoder, and therefore, the block-based motion field can be employed for the frame interpolation at the receiver.

In [40], pixel classification is implemented, and the pixels are interpolated in different ways depending on whether the pixels are unchanged, uncovered or occluded. For those pixels that exist in both the current and previous frames, the linear MCI

technique is used. As for those pixels that belong to an uncovered region, zero motion

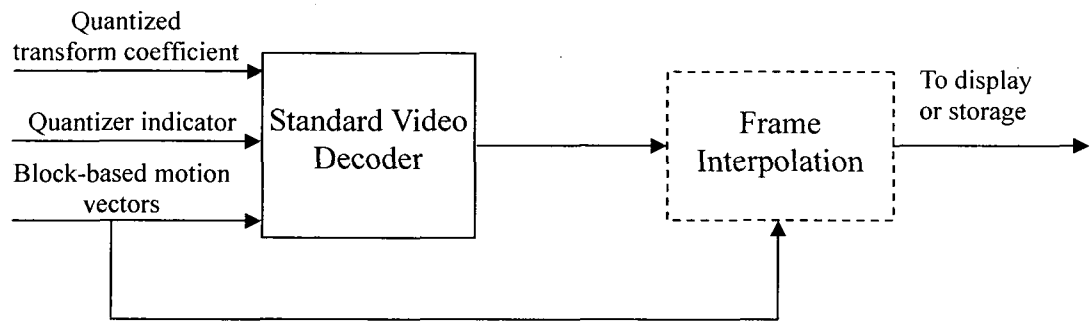


Figure 2.4: Block-based MCI scheme based on conventional video encoding standards.

vectors are employed, based on the assumption that the positions of those pixels can only be found in the current frame, and cannot be compensated by the pixels in the previous frame. Similarly, for those pixels that are in the so-called occluded region, zero vectors are applied, based on the assumption that those pixels can only be compensated with the pixels in the previous frame. Since time-consuming motion estimation is not required at the decoder, this scheme results in significant cost savings. Also, it can provide a relatively better subjective quality compared with a non-motion-compensated interpolation scheme. However, since the pixels in the interpolated frames have to be classified into three classes at the receiver, this process has a high computational complexity.

In [41], to obtain a dense motion vector field, the information carried by the transmitted block-based motion vectors is exploited through a mesh-based mapping. Besides, to get a higher subjective quality for the interpolated frames, this scheme also employs some other techniques such as the foreground/background segmentation and image change detection. Obviously, the computational complexity of this scheme is too high to be implemented in an inexpensive video decoder. As pointed out in [42], one

major difficulty in a block-based MCI scheme is in dealing with the overlapped pixels and unfilled holes, which are left after performing the block-based motion compensation for the interpolated frame. Spatial interpolation may be used to deal with these holes, but it is quite difficult, since the neighborhood of a hole may still contain other holes. In view of this, the authors in [42] have proposed a scheme, the deformable block-based frame MCI (DB-FMCI) scheme, wherein only those holes that follow some predefined simple patterns can be dealt with in conjunction with pixel classification. The computational complexity of this scheme prevents it from being implemented in an inexpensive decoder.

In [43], a block-based MCI algorithm called the scaled motion vector field (SMVF) algorithm is proposed, and is intended to be used in an inexpensive video decoder. In this algorithm, block-based motion vectors are used instead of the pixel-based motion vectors, and pixel classification is not carried out. Although its computational complexity is relatively low, the visual quality of the interpolated frames is not good enough, as blur artifacts appear due to the smoothing technique employed in the algorithm to handle the overlapped pixels. In [44], the Inertia-MCI (I-MCI) algorithm, which exploits the block-based motion information and employs inertia motion prediction at the decoder, is proposed. This algorithm has been claimed to provide better subjective and objective qualities for the interpolated frames than that provided by the other algorithms. However, the algorithm carries out a pixel classification procedure that cannot be performed until the succeeding frame arrives, thus resulting in a significant delay in reconstructing the interpolated frame; hence, this procedure is not practical for real-time video applications. In addition, some post-processing steps, such as the fine

scene-segmentation, are needed, and this would result in an increased computational complexity. It is also to be pointed out that in this scheme the unfilled pixels are predicted by frame repetition, and this method may produce stripe artifacts.

In summary, few of the existing block-based MCI algorithms can work without pixel classification, which requires a computationally expensive image segmentation process. Besides, in view of the difficulty of employing spatial interpolation to deal with the holes, some of the existing schemes employ simple techniques, such as the pixel averaging or repetition, to fill the holes. An averaging technique is often used to handle the overlapped pixels, resulting in images that may suffer from blur or stripe effects. An MCI scheme, that can produce a reconstructed frame with a high visual quality at a relatively low computational complexity, would thus be desirable.

As discussed earlier, interpolation errors are due to the overlapped pixels or unfilled holes. As for the overlapped pixels, most block-based MCI schemes use an averaging algorithm to deal with them. With respect to the unfilled holes, either a frame repetition technique or a median filter has been employed to fill these holes [45], [46]. A variety of error concealment techniques [51]-[62], making use of spatial interpolation, have been widely employed for video communication in an error-prone environment, sparse data reconstruction, and image up-conversion. However, none of these error concealment schemes is suitable for the problem at hand. It is to be noted there is one aspect that is common in all these applications: the shape of the region that needs to be interpolated is either known or follows a certain pattern. With respect to the error concealment for video communication, spatial interpolation is used to mask the effect of missing blocks, having fixed shape and size. As far as sparse data reconstruction and

image up-conversion are concerned, once the scale ratio is determined, the position for the interpolated pixels has to follow a certain pattern. However, in an MCI scheme, the holes can have an arbitrary shape, which is determined by the motion vectors available at the receiver, and varies from frame to frame. Furthermore, the spatial neighborhood of a hole may still contain other holes. This difficulty has also been pointed out in [42], and as a consequence, the authors have taken an approach that is different from that of spatial interpolation to resolve the issue. To the best of our knowledge, no MCI scheme employing a spatial interpolation technique has been proposed to deal with the overlapped or unfilled pixels. Therefore, finding a mathematical approach and deriving a formula to conceal the interpolation errors still remains as challenges to block-based MCI schemes.

2.3 Fundamentals of Finite Element Method

Finite element method (FEM) is a powerful tool for obtaining numerical solutions to a wide variety of engineering problems, such as those in mechanical, aeronautical, biomechanical, and automotive engineering. The basic idea behind FEM is that a given structure is divided into a number of smaller elements having finite dimensions, called *finite elements*. As a result, the original structure can then be considered as a collection of these elements, which are connected at a finite number of nodal points, called *nodes*. The properties of the various elements are formulated, and then combined in a certain way, so that the properties of the entire structure can be obtained. Thus, instead of solving the problem for the entire structure in one operation, the FEM focuses on the formulation of

the properties of the constituent elements of the structure. The basic steps involved in the FEM are as follows.

1. Segment the given structure into a number of non-overlapping finite elements of simple geometry.
2. Express the value of any point in an element as a function of the values at its nodes, the number of nodes being dependent on the geometry of the elements.
3. Approximate the response of the structure based on the response of the discrete model obtained by connecting or assembling the collection of all the elements in a way that continuity is ensured at each node. The necessary boundary conditions are imposed and the equations of equilibrium are then solved for the primary unknowns.

To have a better understanding of the basic principles and procedures of the method, we apply the FEM to an elementary problem. In the following example, we employ FEM to examine the steady state heat conduction in a plane. The two-dimensional steady-state heat conduction problem can be stated as follows. Consider the region R bounded by the curve C in the (x, y)-plane, as shown in Fig. 2.5. The task is to find the temperature function $T(x, y)$ in R, which satisfies the partial differential equation

$$\nabla \cdot \left(\nabla T(x, y)\right) = 0 \tag{2.6}$$

where ∇ is the two-dimensional vector differential operator

$$\nabla(\cdot) = i\frac{\partial}{\partial x}(\cdot) + j\frac{\partial}{\partial y}(\cdot)$$

This equation is subject to the boundary conditions on C, which can be divided into two parts C_1 and C_2,

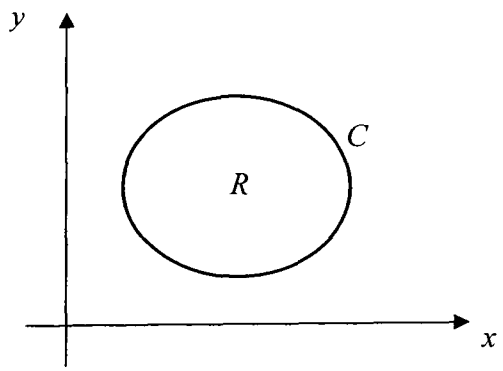


Figure 2.5: Heat conduction region R for the temperature function $T(x,y)$

$$T(x,y) = g(x,y) \text{ on } C_1 \tag{2.7}$$

and

$$\nabla T(x,y) \cdot \vec{n} = 0 \text{ on } C_2 \tag{2.8}$$

where $\vec{n}$ is a unit vector normal to C and directed out of R. Now let us convert (2.6), together with (2.7) and (2.8) into a variational formulation. We proceed by multiplying the left side of (2.6) by δT and integrating over R, and obtain

$$\int_R \delta T \, \nabla \cdot \left(\nabla T(x,y)\right) dR = 0 \tag{2.9}$$

Based on vector identity, (2.9) can be written as

$$\int_R \delta T \, \nabla \cdot \left(\nabla T(x,y)\right) dR = \int_R \left[\nabla \cdot \left(\delta T \, \nabla T(x,y)\right) - \frac{1}{2} \delta \left(\nabla T(x,y)\right)^2 \right] dR \tag{2.10}$$

If we use the divergence theorem, and require δT be zero on C_1, the above is equivalent to

$$\int_R \delta T \, \nabla \cdot \left(\nabla T(x,y)\right) dR = -\frac{1}{2} \int_R \delta \left(\nabla T(x,y)\right)^2 dR \tag{2.11}$$

Therefore, the variational formation of the boundary value problem of (2.6) is

$$\delta I = 0 \tag{2.12}$$

where

$$I = \iint_R \left[\frac{1}{2} (\nabla T(x,y))^2 \right] dR \tag{2.13}$$

Let the region R be divided into N non-overlapping triangles as shown in Fig. 2.6. These triangles are *finite elements* of the region R. In order to seek a solution to (2.6), we employ an approach, through the use of which, the temperature function $T(x,y)$ can

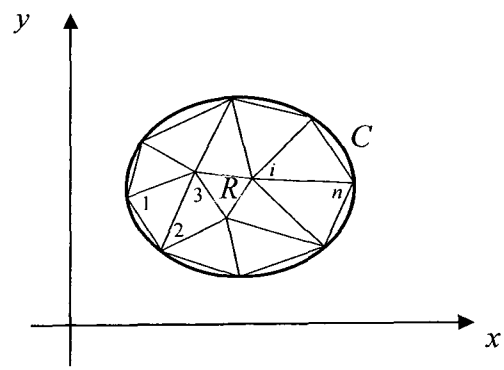


Figure 2.6: Triangular element segmentation of the region R

be represented by its values at the nodes of the triangles throughout R. Let the nodes of the triangles be numbered globally from 1 to n, then the temperature distribution in R will be represented by the array

$$[T]^T = [T_1, T_2, T_3, \cdots, T_n] \tag{2.14}$$

A typical triangular element with its nodal coordinates is presented in Fig. 2.7. Let us introduce the shape function $N_i^e(x,y)$, $i = 1, 2, 3$. $N_i^e(x,y)$ is linear in both x and y on e, and is zero outside e. Also, $N_i^e(x,y)$ has a value of 1 at node i, and a value 0 at

node j, where $j \neq i$. Therefore, the temperature function $T(x,y)$ on e can be expressed as

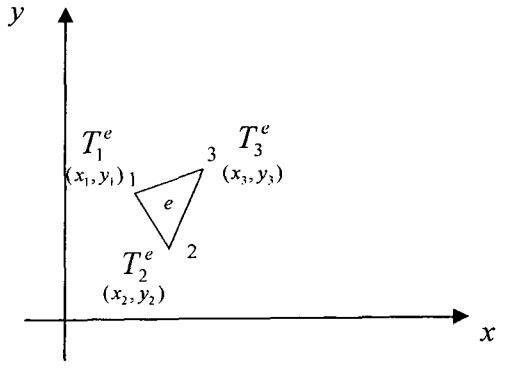


Figure 2.7: A typical triangular element with its nodal coordinates

$$T^e(x,y) = T_1^e N_1^e(x,y) + T_2^e N_2^e(x,y) + T_3^e N_3^e(x,y)$$
$$= [N_1^e(x,y) \quad N_2^e(x,y) \quad N_3^e(x,y)] \begin{bmatrix} T_1^e \\ T_2^e \\ T_3^e \end{bmatrix} \tag{2.15}$$

with

$$N_1^e(x,y) = \frac{1}{2s} \begin{vmatrix} 1 & x & y \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{vmatrix} \tag{2.16}$$

$$N_2^e(x,y) = \frac{1}{2s} \begin{vmatrix} 1 & x_1 & y_1 \\ 1 & x & y \\ 1 & x_3 & y_3 \end{vmatrix} \tag{2.17}$$

$$N_3^e(x,y) = \frac{1}{2s} \begin{vmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x & y \end{vmatrix} \tag{2.18}$$

where s denotes the area of element e, that is

$$s = \frac{1}{2}\begin{vmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{vmatrix} \tag{2.19}$$

Since the region R has been divided into N elements , I can be written in the form

$$I = \sum_{e=1}^{N} I^e \tag{2.20}$$

where

$$I^e = \frac{1}{2}\int_e [(\nabla T^e)^2] dR \tag{2.21}$$

It can be shown that [66]

$$\nabla T^e = \begin{bmatrix} \dfrac{\partial N_1^e(x,y)}{\partial x} & \dfrac{\partial N_2^e(x,y)}{\partial x} & \dfrac{\partial N_3^e(x,y)}{\partial x} \\ \dfrac{\partial N_1^e(x,y)}{\partial y} & \dfrac{\partial N_2^e(x,y)}{\partial y} & \dfrac{\partial N_3^e(x,y)}{\partial y} \end{bmatrix} \begin{bmatrix} T_1^e \\ T_2^e \\ T_3^e \end{bmatrix} \tag{2.22}$$

Let the coefficient matrix be denoted by $[B^e]$, that is,

$$[B^e] = \begin{bmatrix} \dfrac{\partial N_1^e(x,y)}{\partial x} & \dfrac{\partial N_2^e(x,y)}{\partial x} & \dfrac{\partial N_3^e(x,y)}{\partial x} \\ \dfrac{\partial N_1^e(x,y)}{\partial y} & \dfrac{\partial N_2^e(x,y)}{\partial y} & \dfrac{\partial N_3^e(x,y)}{\partial y} \end{bmatrix} \tag{2.23}$$

Then, we have a compact form for $(\nabla T^e)^2$

$$(\nabla T^e)^2 = [T^e]^T [\kappa^e][T^e] \tag{2.24}$$

where $[\kappa^e]$ is defined as

$$[\kappa^e] = [B^e]^T [B^e] \tag{2.25}$$

and $[T^e]$ is given by

$$\left[T^e\right]^T = \begin{bmatrix} T_1^e & T_2^e & T_3^e \end{bmatrix} \tag{2.26}$$

Hence, (2.21) can be expressed in the following form

$$I^e = s\left[T^e\right]^T \left[\kappa^e\right]\left[T^e\right] \tag{2.27}$$

Let us assemble all the N elements in region R, assuming the local nodes 1, 2 and 3 of the element e correspond to the global nodes I, J and K,

$$T_1^e = T_I,\ T_2^e = T_J,\ T_3^e = T_K \tag{2.28}$$

We now recall that the region R is divided into N elements with a total of n nodes; hence,

$$[T]^T = \begin{bmatrix} T_1 & \cdots & T_I & \cdots & T_J & \cdots & T_K & \cdots & T_n \end{bmatrix} \tag{2.29}$$

If we define an incidence matrix $\left[\Lambda^e\right]$ by

$$\left[\Lambda^e\right]^T = \begin{array}{c} \begin{matrix} 0 & \cdots & I & \cdots & J & \cdots & K & \cdots & n \end{matrix} \\ \begin{bmatrix} 0 & \cdots & 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ 0 & \cdots & 0 & \cdots & 1 & \cdots & 0 & \cdots & 0 \\ 0 & \cdots & 0 & \cdots & 0 & \cdots & 1 & \cdots & 0 \end{bmatrix} \end{array} \tag{2.30}$$

then

$$\left[T^e\right] = \left[\Lambda^e\right]^T [T] \tag{2.31}$$

Therefore, (2.26) can be written as

$$I^e = \frac{1}{2}\left[T\right]^T \left[K^e\right][T] \tag{2.32}$$

where

$$\left[K^e\right] = s \cdot \left[\Lambda^e\right]\left[\kappa^e\right]\left[\Lambda^e\right] \tag{2.33}$$

Then, by substituting (2.32) into (2.20), we have

$$I = \frac{1}{2}\left[T\right]^T [K][T] \tag{2.34}$$

where $[K]$ is defined as

$$[K]=\sum_{e=1}^{N}[K^e] \tag{2.35}$$

As a matter of fact, satisfying $\delta I=0$ is equivalent to satisfying $\frac{\partial I}{\partial T_i}=0$, $i=1,\cdots,n$.

Thus, we have

$$[K][T]=0 \tag{2.36}$$

Hence, through the use of the finite element method, the boundary value problem given by (2.6) to (2.8) is converted into a system of algebraic equations, the solutions to which can be solved in the interior of R. This example has illustrated how the FEM method can be employed to solve complex engineering problems. In Chapter 3, we will apply the finite element method to design a generalized image interpolation model, which can be utilized for both the spatial resolution enhancement of images and the temporal resolution enhancement of encoded video sequences.

2.4 Summary

This chapter has introduced some basic concepts related to edge-directed spatial resolution enhancement and block-based MCI schemes for temporal resolution enhancement of encoded video sequences. A brief literature review regarding the existing edge-directed spatial resolution enhancement schemes and block-based MCI schemes, along with an analysis of the drawbacks of these schemes has been carried out. This has provided the motivation for designing an appropriate interpolation model. Since the finite element method is chosen for the development of such an interpolation model with the

purpose of devising an image interpolation technique, fundamentals of FEM have been introduced. An example of applying FEM to solve the steady state heat conduction problem has also been given, in order to present the basic principles and steps of applying the finite element method. The advantage of the FEM in providing simplicity of piecewise approximation of a function given its values at discrete points, has also been highlighted.

Chapter 3

A Generalized Image Interpolation Model Based on Finite Element Method

3.1 Introduction

As discussed earlier, in the existing edge-directed spatial resolution enhancement schemes, an unknown pixel is interpolated based on the estimation made only with the neighboring known pixels. Consequently, there are two outstanding issues in such schemes.

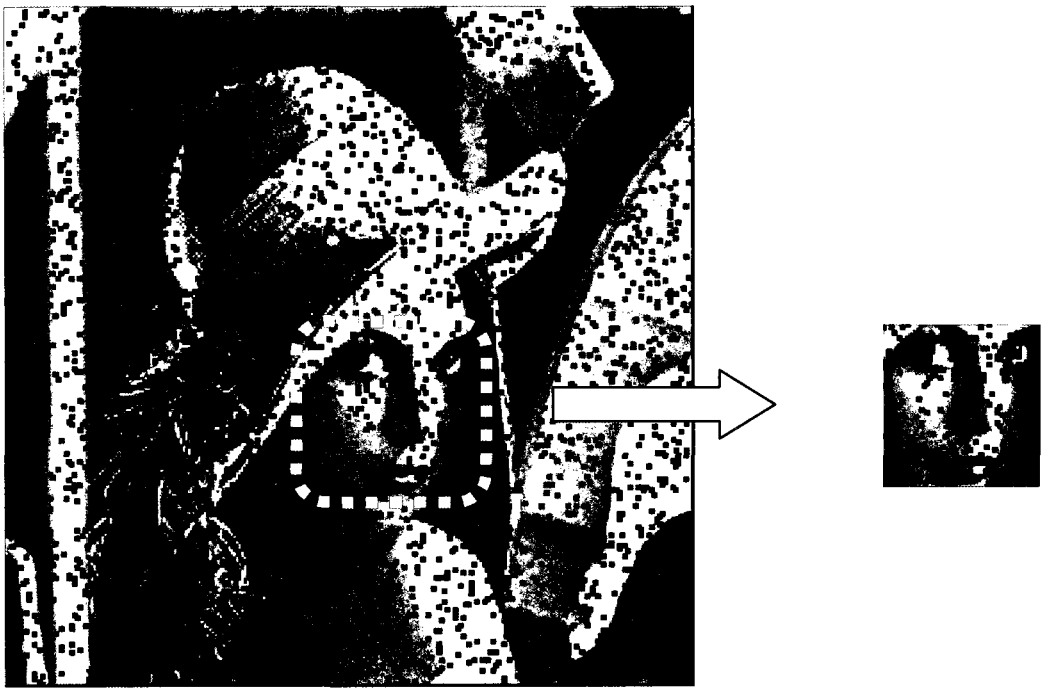
1. They are not capable of handling an arbitrary magnification factor
2. They cannot complete the interpolation process without multiple interpolation steps, even when up-scaling an image by a factor of two

To resolve these issues, we need to develop a new interpolation model, which is not restricted to making use of the neighboring pixels with known values, but also takes into account the neighboring pixels with unknown values in the interpolation process. As for temporal resolution enhancement, an ideal interpolation model is one which is capable of handling an area having an arbitrary shape, which may contain overlapped and/or unfilled pixels.

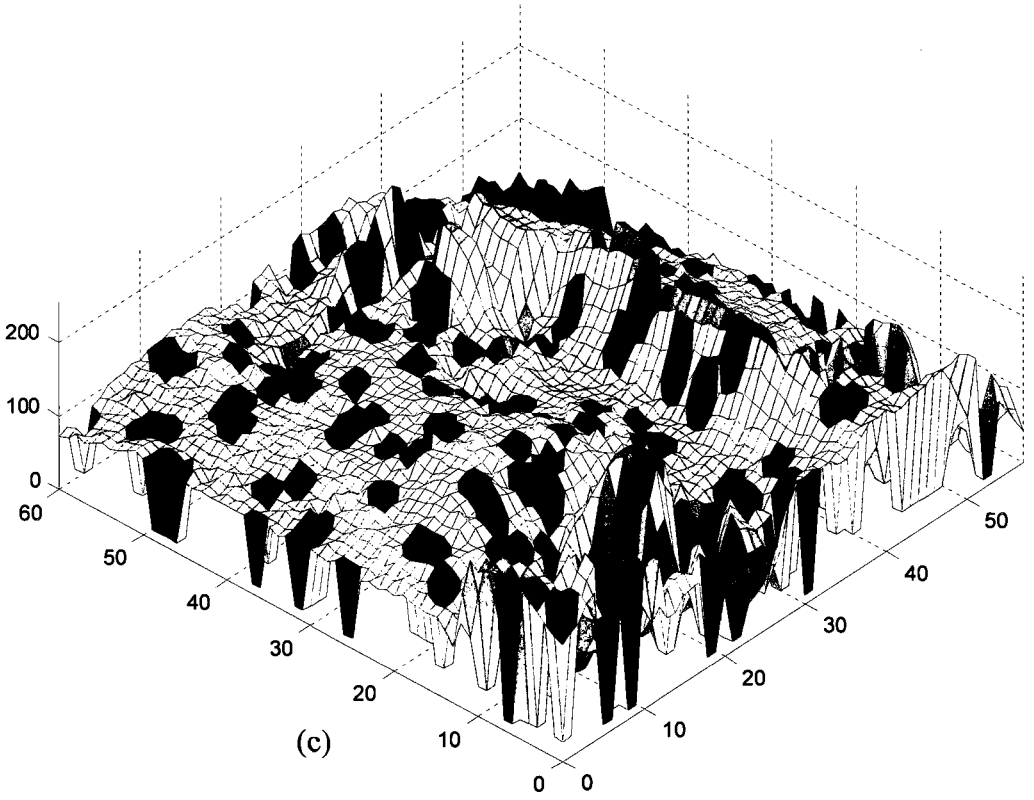
In view of these requirements for spatial and temporal resolution enhancement, there is a need to develop a new interpolation model that is able to interpolate unknown pixels under the following three conditions.

1. The spatial neighborhood of an unknown pixel contains known pixels as well as unknown pixels
2. A collection of the unknown pixels has an arbitrary shape
3. The interpolation for a certain unknown pixel involves the neighboring pixels with both known and unknown values

To meet the above requirements, a generalized image interpolation model based on FEM is presented in this chapter. To have a better understanding of the problem at hand, consider the test image *Lena* with a number of missing pixels, as illustrated in Fig. 3.1(a). The *holes*, which are deliberately generated as collections of unknown pixels, do not have a fixed pattern, and are arbitrarily shaped. In Fig. 3.1(b), the discontinuities due to the presence of the holes can be seen in the 3D surface plot of the face portion of the image. In Fig. 3.2 (a), the original test image *Lena* and the face portion of the image are shown; the 3D surface plot of the face portion sub-image is illustrated in Fig. 3.2(b). Our goal is to design an interpolation model, so that each unknown pixel is interpolated by a proper value, such that the continuity of the 3D surface plot of the resulting image can be restored as close as possible to that shown in Fig. 3.2(b). As pointed out in [42] and [47], there are two aspects that make this a difficult task: (i) the holes have arbitrary shapes, and (ii) the neighborhood of a hole may still contain other holes. To accomplish this difficult task, we develop a FEM-based interpolation model, which can interpolate unknown pixels in an image under the conditions mentioned earlier.

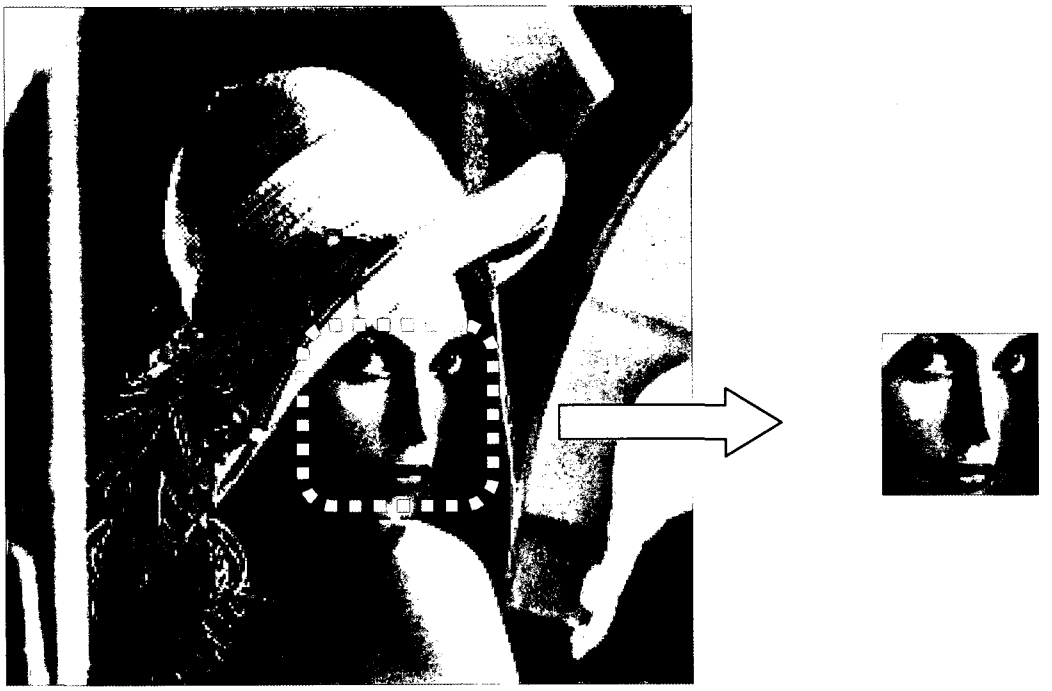


(a) (b)

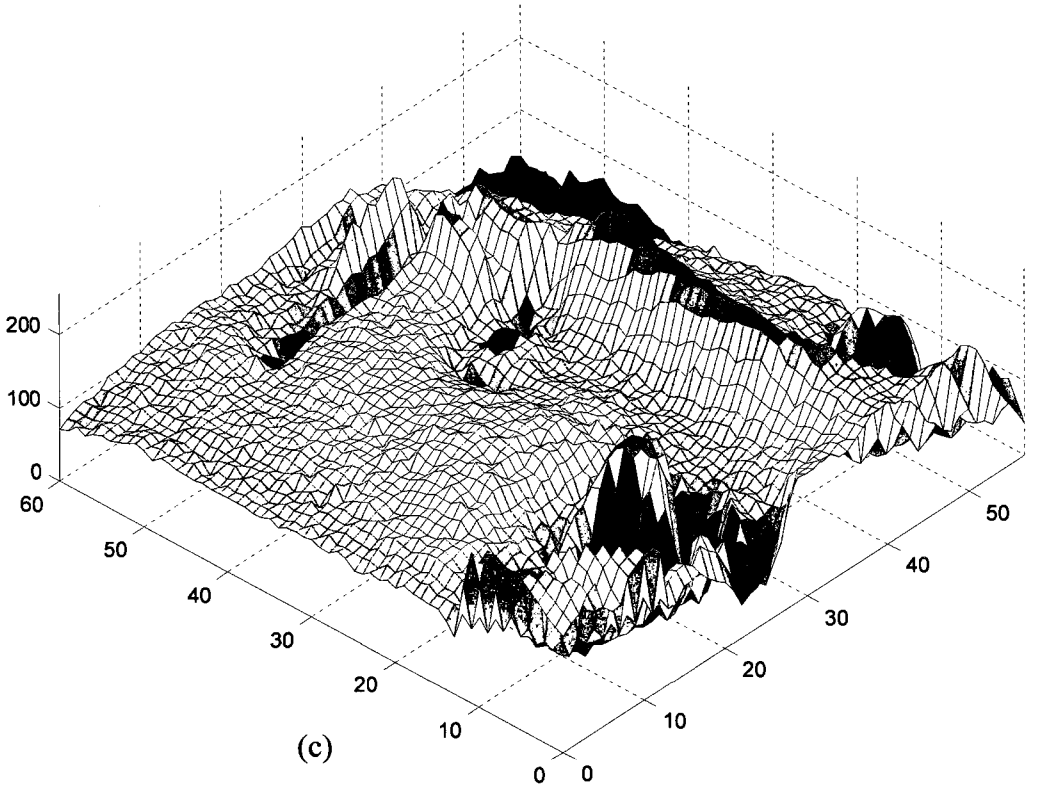


(c)

Figure 3.1: (a) Test image *Lena* manipulated with arbitrarily-shaped holes. (b) A sub-image from the face portion of the image in (a). (c) The 3D surface plot of the sub-image in (b)



(a) (b)



(c)

Figure 3.2: (a) Original test image *Lena.* (b) A sub-image from the face portion of the image in (a). (c) The 3D surface plot of the sub-image in (b)

3.2 Finite Element Method-Based Image Interpolation Model

In order to interpolate unknown pixels in an image, let us apply the FEM [63], [64] to construct a continuous luminance function $L(x, y)$. The domain Ω of this function is an image I_{mg}, in which unknown pixels are randomly present as shown in Fig 3.1(a). The constraint we set for this function is that an interpolated pixel should fit smoothly with its immediate-neighboring pixels, which are either known pixels or themselves interpolated pixels.

3.2.1 Segmentation of the domain into non-overlapping elements

Let us consider the image I_{mg} to have $(W \times H)$ pixels, and the domain Ω to be

$$\Omega = [0, W-1] \times [0, H-1] \tag{3.1}$$

We employ triangular segmentation and partition Ω into $N (N \in Z)$ non-overlapping triangular elements e_ν, ($\nu = 1, \ldots, N$), the nodes of which correspond to the pixel locations in I_{mg} as shown in Fig. 3.3(a). These elements may be classified into two types of local triangular elements, Type I and Type II elements, the first, second, and third nodes of which are ordered counterclockwise (see Figs. 3.3(b) and 3.3(c)). D is the set containing all the nodes in Ω, that is $D \subset \Omega$, or

$$D = (0,1, \ldots, W-1) \times (0,1, \ldots, H-1) \tag{3.2}$$

After the triangular segmentation of Ω, the nodes corresponding to the known pixels are referred to as the known nodes, whereas the remaining nodes as unknown nodes. Let $\mathrm{K} \subset D$ denote the set of known nodes and $U \subset D$ the set of unknown nodes; $U \bigcup \mathrm{K} = D$ and $U \bigcap \mathrm{K} = \varnothing$. Hence, $L(x, y)$ has known values at $(x, y) \in \mathrm{K}$, and these

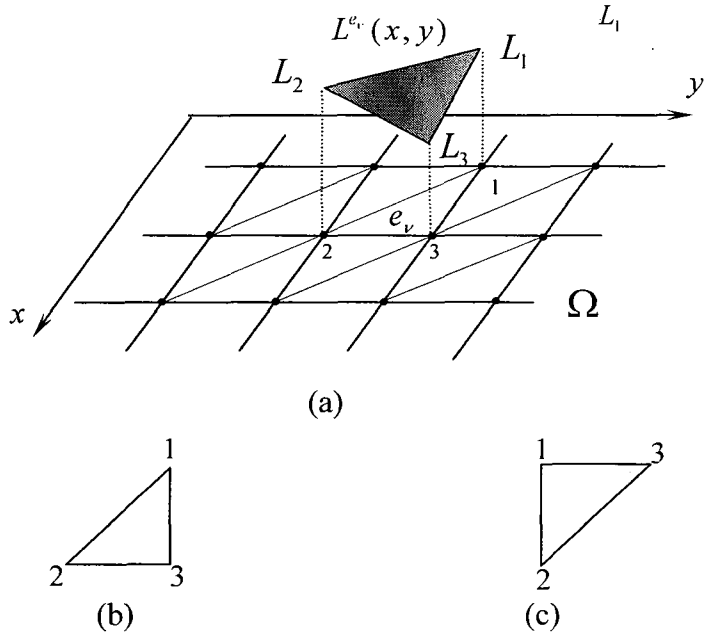


Figure 3.3: (a) The triangular elements partitioned in the domain Ω of $L(x,y)$. $L^{e_\nu}(x,y)$ represents the continuous function for the element e_ν. (b) Type I local triangular element. (c) Type II local triangular element

values are not altered in the interpolation process.

3.2.2 Response of each element in terms of the values at its nodes

By applying the FEM, $L^{e_\nu}(x,y)$ can be expressed as a function of the values of the response at its nodes in the element e_ν. This process can be summarized as follows.

$$\begin{aligned} &L(x,y) = L^{e_\nu}(x,y) \quad (x,y) \in e_\nu \\ &L^{e_\nu}(x,y) = L_1^{e_\nu}\varphi_1(x,y) + L_2^{e_\nu}\varphi_2(x,y) + L_3^{e_\nu}\varphi_3(x,y) \end{aligned} \tag{3.3}$$

where $L^{e_\nu}(x,y)$ $\left((x,y) \in e_\nu\right)$ is a continuous function as shown in Fig. 3.3(a), $L_i^{e_\nu} = L^{e_\nu}(x_i,y_i)$, $(i=1,2,3)$, and $(x_i,y_i) \in D$, (x_i,y_i) being the location of the *ith* node in e_ν. The shape functions are given by

$$\varphi_1(x,y)=\frac{1}{2s}\begin{vmatrix}1 & x & y\\ 1 & x_2 & y_2\\ 1 & x_3 & y_3\end{vmatrix}$$

$$\varphi_2(x,y)=\frac{1}{2s}\begin{vmatrix}1 & x_1 & y_1\\ 1 & x & y\\ 1 & x_3 & y_3\end{vmatrix} \tag{3.4}$$

$$\varphi_3(x,y)=\frac{1}{2s}\begin{vmatrix}1 & x_1 & y_1\\ 1 & x_2 & y_2\\ 1 & x & y\end{vmatrix}$$

with s denoting the area of the element e_ν.

3.2.3 Approximation of the values for the unknown nodes

To satisfy the requirements set earlier regarding the smooth fit of an interpolated pixel, we apply the optimization constraint that the surface represented by $L(x,y)$ has the smallest area. Then, we can find suitable luminance values for the unknown nodes to minimize the function

$$\iint_\Omega \sqrt{1+\left(\frac{\partial L(x,y)}{\partial x}\right)^2+\left(\frac{\partial L(x,y)}{\partial y}\right)^2}\,dxdy \tag{3.5}$$

This is equivalent to finding a set containing $L(i,j)$ at each node $(i,j)\in U$, which minimizes

$$\sum_{\nu=1}^{N}\iint_{e_\nu}\left[\left(\frac{\partial L^{e_\nu}(x,y)}{\partial x}\right)^2+\left(\frac{\partial L^{e_\nu}(x,y)}{\partial y}\right)^2\right]dxdy \tag{3.6}$$

Let us define a variational function

$$a(u(x,y),v(x,y))=\iint_\Omega \frac{\partial u(x,y)}{\partial x}\frac{\partial v(x,y)}{\partial x}+\frac{\partial u(x,y)}{\partial y}\frac{\partial v(x,y)}{\partial y}dxdy \tag{3.7}$$

Then, we have

$$\sum_{\nu=1}^{N} a(L^{e_\nu}(x,y), L^{e_\nu}(x,y)) = \sum_{\nu=1}^{N} \iint_{e_\nu} \left[\left(\frac{\partial L^{e_\nu}(x,y)}{\partial x} \right)^2 + \left(\frac{\partial L^{e_\nu}(x,y)}{\partial y} \right)^2 \right] dxdy \tag{3.8}$$

where

$$\begin{aligned} a(L^{e_\nu}(x,y), L^{e_\nu}(x,y)) = {} & L_1^{e_\nu} L_1^{e_\nu} a(\varphi_1(x,y), \varphi_1(x,y)) + L_2^{e_\nu} L_2^{e_\nu} a(\varphi_2(x,y), \varphi_2(x,y)) \\ & + L_3^{e_\nu} L_3^{e_\nu} a(\varphi_3(x,y), \varphi_3(x,y)) + 2L_1^{e_\nu} L_2^{e_\nu} a(\varphi_1(x,y), \varphi_2(x,y)) \\ & + 2L_1^{e_\nu} L_3^{e_\nu} a(\varphi_1(x,y), \varphi_3(x,y)) + 2L_2^{e_\nu} L_3^{e_\nu} a(\varphi_2(x,y), \varphi_3(x,y)) \end{aligned}$$

From (3.4), the following can be obtained,

$$a(\varphi_1(x,y), \varphi_1(x,y)) = \frac{1}{4s}\left[(y_2 - y_3)^2 + (x_3 - x_2)^2 \right] \tag{3.9}$$

$$a(\varphi_1(x,y), \varphi_2(x,y)) = \frac{1}{4s}\left[(y_2 - y_3)(y_3 - y_1) + (x_3 - x_2)(x_1 - x_3) \right] \tag{3.10}$$

$$a(\varphi_1(x,y), \varphi_3(x,y)) = \frac{1}{4s}\left[(y_2 - y_3)(y_1 - y_2) + (x_2 - x_1)(x_3 - x_2) \right] \tag{3.11}$$

$$a(\varphi_2(x,y), \varphi_2(x,y)) = \frac{1}{4s}\left[(y_3 - y_1)^2 + (x_1 - x_3)^2 \right] \tag{3.12}$$

$$a(\varphi_2(x,y), \varphi_3(x,y)) = \frac{1}{4s}\left[(y_3 - y_1)(y_1 - y_2) + (x_1 - x_3)(x_2 - x_1) \right] \tag{3.13}$$

$$a(\varphi_3(x,y), \varphi_3(x,y)) = \frac{1}{4s}\left[(y_1 - y_2)^2 + (x_2 - x_1)^2 \right] \tag{3.14}$$

Hence, for Type I triangular element, we have

$$\begin{cases} a(\varphi_1(x,y), \varphi_1(x,y)) = \dfrac{1}{2} \\ a(\varphi_1(x,y), \varphi_2(x,y)) = 0 \\ a(\varphi_1(x,y), \varphi_3(x,y)) = -\dfrac{1}{2} \\ a(\varphi_2(x,y), \varphi_2(x,y)) = \dfrac{1}{2} \\ a(\varphi_2(x,y), \varphi_3(x,y)) = -\dfrac{1}{2} \\ a(\varphi_3(x,y), \varphi_3(x,y)) = 1 \end{cases} \tag{3.15}$$

As for Type II triangular element, the following can be obtained.

$$\begin{cases} a(\varphi_1(x,y),\varphi_1(x,y)) = 1 \\ a(\varphi_1(x,y),\varphi_2(x,y)) = -\dfrac{1}{2} \\ a(\varphi_1(x,y),\varphi_3(x,y)) = -\dfrac{1}{2} \\ a(\varphi_2(x,y),\varphi_2(x,y)) = \dfrac{1}{2} \\ a(\varphi_2(x,y),\varphi_3(x,y)) = 0 \\ a(\varphi_3(x,y),\varphi_3(x,y)) = \dfrac{1}{2} \end{cases} \tag{3.16}$$

To minimize (3.8), we set

$$\begin{cases} \dfrac{\partial a(L^{e_v}(x,y),L^{e_v}(x,y))}{\partial L_1^{e_v}} = 0 \\ \dfrac{\partial a(L^{e_v}(x,y),L^{e_v}(x,y))}{\partial L_2^{e_v}} = 0 \\ \dfrac{\partial a(L^{e_v}(x,y),L^{e_v}(x,y))}{\partial L_3^{e_v}} = 0 \end{cases} \tag{3.17}$$

By using (3.3), (3.17) can be expressed as

$$\begin{cases} 2L_1^{e_v} a(\varphi_1(x,y),\varphi_1(x,y)) + 2L_2^{e_v} a(\varphi_1(x,y),\varphi_2(x,y)) + 2L_3^{e_v} a(\varphi_1(x,y),\varphi_3(x,y)) = 0 \\ 2L_2^{e_v} a(\varphi_2(x,y),\varphi_2(x,y)) + 2L_1^{e_v} a(\varphi_1(x,y),\varphi_2(x,y)) + 2L_3^{e_v} a(\varphi_2(x,y),\varphi_3(x,y)) = 0 \\ 2L_3^{e_v} a(\varphi_3(x,y),\varphi_3(x,y)) + 2L_1^{e_v} a(\varphi_1(x,y),\varphi_3(x,y)) + 2L_2^{e_v} a(\varphi_2(x,y),\varphi_3(x,y)) = 0 \end{cases}$$

..... (3.18)

Hence, the local nodal equation can be expressed as

$$2\begin{pmatrix} a(\varphi_1,\varphi_1) & a(\varphi_1,\varphi_2) & a(\varphi_1,\varphi_3) \\ a(\varphi_2,\varphi_1) & a(\varphi_2,\varphi_2) & a(\varphi_2,\varphi_3) \\ a(\varphi_3,\varphi_1) & a(\varphi_3,\varphi_2) & a(\varphi_3,\varphi_3) \end{pmatrix}\begin{pmatrix} L_1^{e_v} \\ L_2^{e_v} \\ L_3^{e_v} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \tag{3.19}$$

From (3.19), (3.15) and (3.16), we have, for Type I element

$$\begin{pmatrix} 1 & 0 & -1 \\ 0 & 1 & -1 \\ -1 & -1 & 2 \end{pmatrix} \begin{pmatrix} L_1^{e_v} \\ L_2^{e_v} \\ L_3^{e_v} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \tag{3.20}$$

while for Type II triangular element, we have

$$\begin{pmatrix} 2 & -1 & -1 \\ -1 & 1 & 0 \\ -1 & 0 & 1 \end{pmatrix} \begin{pmatrix} L_1^{e_v} \\ L_2^{e_v} \\ L_3^{e_v} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \tag{3.21}$$

Let us define

$$\left[L^{e_v}\right]^T = \left[L_1^{e_v}, L_2^{e_v}, L_3^{e_v}\right] \tag{3.22}$$

Then, (3.20) and (3.21) can be written in the following form

$$\left[k^{e_v}\right]\left[L^{e_v}\right] = 0 \tag{3.23}$$

where $\left[k^{e_v}\right]$ is the coefficient matrix for local nodal equations.

After the local nodal equations for each triangular element are derived, we can proceed to add up their contributions in order to form the global nodal equations. Let us cover all the nodes in the set D in a regular raster scan so that all the $n = W \times H$ nodes can be ordered as

$$[L]^T = [L_1, L_2, L_3, \ldots, L_n] \tag{3.24}$$

For instance, for the Type I triangular element e_v, the luminance value at the local nodes 1, 2, and 3 are equal to the luminance value at the global nodes I, J, and K respectively. That is,

$$L_1^{e_v} = L_I, L_2^{e_v} = L_J, L_3^{e_v} = L_K \tag{3.25}$$

By considering all the nodes ordered as per (3.24), the local node equation shown in (3.20) has a contribution to the global nodal equations as shown below.

$$\begin{array}{c} \\ \vdots \\ I \\ \vdots \\ J \\ \vdots \\ \mathrm{K} \\ \vdots \end{array}\begin{array}{c} \begin{array}{ccccccc} \cdots & I & \cdots & J & \cdots & K & \cdots \end{array} \\ \begin{pmatrix} & \vdots & & \vdots & & \vdots & \\ \cdots & 1 & \cdots & 0 & \cdots & -1 & \cdots \\ & \vdots & & \vdots & & \vdots & \\ \cdots & 0 & \cdots & 1 & \cdots & -1 & \cdots \\ & \vdots & & \vdots & & \vdots & \\ \cdots & -1 & \cdots & -1 & \cdots & 2 & \cdots \\ & \vdots & & \vdots & & \vdots & \end{pmatrix} \end{array}\begin{pmatrix} \vdots \\ L_I \\ \vdots \\ L_J \\ \vdots \\ L_K \\ \vdots \end{pmatrix} = \begin{pmatrix} \vdots \\ 0 \\ \vdots \\ 0 \\ \vdots \\ 0 \\ \vdots \end{pmatrix} \tag{3.26}$$

It can be seen from (3.26) that the nodes $L_1^{e_\nu}$, $L_2^{e_\nu}$, and $L_3^{e_\nu}$ in element e_ν act as L_I, L_J and L_K respectively in the global nodal equations. At the same time, the coefficient matrices from the local equations contribute to the coefficient matrix for the global nodal equations. Recalling the incidence matrix defined in (2.30), we can relate the luminance values at the local nodes and global nodes as

$$\left[L^{e_\nu}\right] = \left[\Lambda^{e_\nu}\right]^T [L] \tag{3.27}$$

equation (3.26) can be re-written as

$$[\Psi^{e_\nu}][L] = 0 \tag{3.28}$$

where

$$[\Psi^{e_\nu}] = \left[\Lambda^{e_\nu}\right]\left[k^{e_\nu}\right]\left[\Lambda^{e_\nu}\right]^T \tag{3.29}$$

If we continue with this process, the contribution of all the N local elements in Ω can be added to obtain the overall global nodal equations in the form

$$[\Psi][L] = 0 \tag{3.30}$$

where

$$[\Psi] = \sum_{\nu=1}^{N} \left[\Psi^{e_\nu}\right] \tag{3.31}$$

The optimal luminance value of an unknown node at $(i, j) \in U$, can be obtained by solving (3.30). Hence, the following equation holds for an unknown node at $(i, j) \in U$,

$$L(i, j) = S_V(i, j) + S_H(i, j) \tag{3.32}$$

with

$$S_V(i, j) = \begin{cases} \frac{1}{2} L(i+1, j) & if\ i = 0 \\ \frac{1}{2} L(i-1, j) & if\ i = H-1 \\ \frac{1}{4}\left[L(i-1, j) + L(i+1, j)\right] & if\ i \neq 0 or H-1 \end{cases} \tag{3.33}$$

$$S_H(i, j) = \begin{cases} \frac{1}{2} L(i, j+1) & if\ j = 0 \\ \frac{1}{2} L(i, j-1) & if\ j = W-1 \\ \frac{1}{4}\left[L(i, j+1) + L(i, j-1)\right] & if\ j \neq 0 or W-1 \end{cases} \tag{3.34}$$

To determine the value for an unknown node $(i, j) \in U$, it requires solving a set of linear equations, the dimension of which, however, could be large, since it is equal to the total number of nodes. Solving these equations directly is not desirable. It can be shown that the coefficient matrix of such a set of linear equations is diagonal-dominant and non-reducible. Therefore, we can employ an iterative method to obtain solutions to such a set of linear equations, with a guaranteed convergence [63]. Hence, we derive a flexible scheme that is performed in a raster scan order, to obtain the luminance value at the unknown node $(i, j) \in U$, using the following equations in an iterative manner.

$$L^{(k+1)}(i, j) = S_V^{(k)}(i, j) + S_H^{(k)}(i, j) \tag{3.35}$$

with

$$S_V^{(k)}(i,j)=\begin{cases}\frac{1}{2}L^{(k)}(i+1,j) & if\ i=0\\ \frac{1}{2}L^{(k+1)}(i-1,j) & if\ i=H-1\\ \frac{1}{4}\left[L^{(k+1)}(i-1,j)+L^{(k)}(i+1,j)\right] & if\ i\neq 0\ or\ H-1\end{cases}\quad (k\geq 0) \tag{3.36}$$

$$S_H^{(k)}(i,j)=\begin{cases}\frac{1}{2}L^{(k)}(i,j+1) & if\ j=0\\ \frac{1}{2}L^{(k+1)}(i,j-1) & if\ j=W-1\\ \frac{1}{4}\left[L^{(k+1)}(i,j-1)+L^{(k)}(i,j+1)\right] & if\ j\neq 0\ or\ W-1\end{cases}\quad (k\geq 0) \tag{3.37}$$

It can be shown that convergence is guaranteed [65] with such an iterative refinement procedure for any unknown node in D. The procedure will not alter the values of the known nodes involved in (3.35), with $L^{(k+1)}(x,y)=L^{(k)}(x,y),(x,y)\in \mathrm{K}$, whereas the unknown nodes will be refined gradually. When the stop criterion is met, we obtain $\hat{L}(i,j)$, $(i,j)\in U$, as the luminance of a interpolated pixel. When all the unknown nodes in Ω are assigned such values, we have the image, the unknown pixels of which have been successfully interpolated.

3.2.4 Image block size and selection of the initialization value

To apply this image interpolation model, an image I_{mg} with size $W\times H$ pixels is partitioned into blocks for the purpose of distributing the computational load among several blocks. To examine the performance resulting from different block sizes, we divide the test image *Lena* as shown in Fig. 3.1 (256×256 pixels) into non-overlapping image blocks of sizes 8×8 pixels, 16×16 pixels, and 32×32 pixels, respectively. Under the same initialization condition (initialized with zero), simulation results reveal that a

larger block size leads to better qualities; however, the number of basic operations (including those of addition, multiplication, and absolute difference) increases. Results in terms of the PSNR and computational complexity, are given in Table 3.1. It is seen that a higher PSNR value can be obtained at the cost of higher computational complexity. This setting provides a tradeoff between the performance and the computational complexity, according to the requirements of a given application.

Table 3.1: Computational complexity and PSNR resulted from different block size (with test image *Lena*)

Block size ($M \times N$ pixels)	8×8	16×16	32×32
PSNR (dB)	32.32	32.44	32.50
Basic operations	370,613	429,990	448,273

The initialization value $L^{(0)}(i, j)$ does not affect the interpolated image result. It, however, has an influence on the speed of convergence. We choose for $L^{(0)}(i, j)$ three different values, namely, 0, the minimum luminance value and the mean luminance value of the known pixels in an image block. Simulation results, given in Table 3.2, show that initialization with the mean value of the known pixels results in the smallest number of basic operations, leading to a faster convergence.

Table 3.2: Comparison of computational complexity with different initialization values (with image block size16×16)

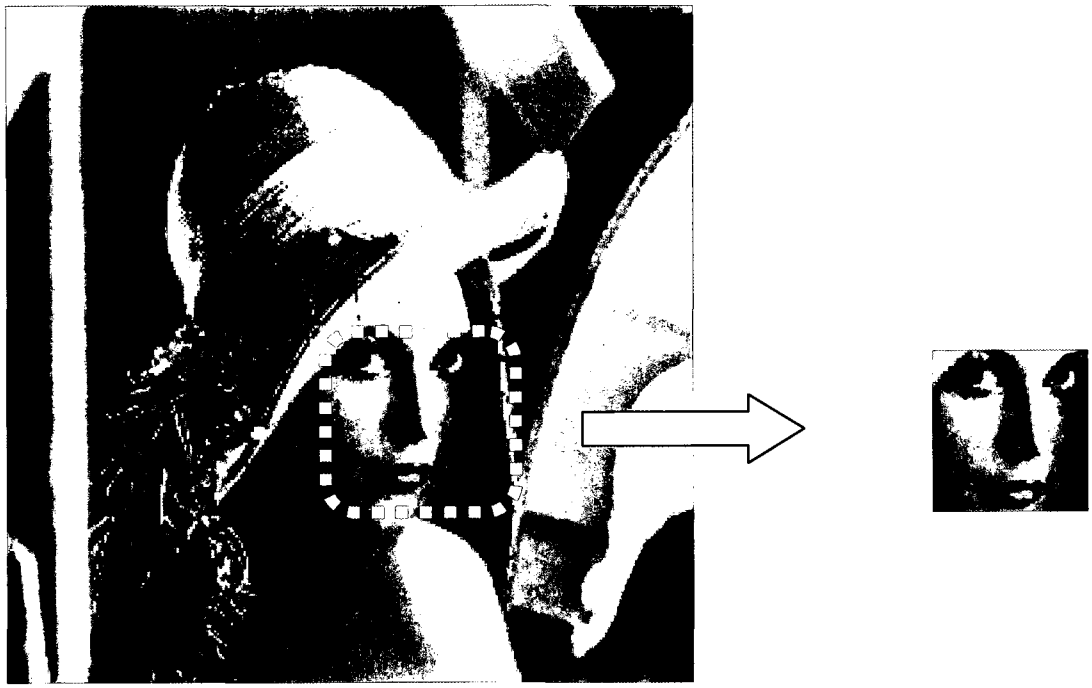
Initialization value	0	Minimum value of the known pixels	Mean value of the known pixels
No. of basic operations	429,990	400,848	296,783

In order to interpolate the missing pixels in the test image *Lena*, a simulation has been carried out. The image block size selected is 16×16, and the initialization value chosen is the mean value of the known pixels. The result of the interpolation is shown in Fig. 3.4. In Fig. 3.4(a) and (b), the interpolated image and the face portion of the image are presented. Fig. 3.4(c) shows the 3D surface plot of the face portion presented in Fig. 3.4(b). From these figures, it can be seen that in the regions, where unknown nodes exist, the interpolated pixels exhibit a smooth variation between the known pixels and the interpolated pixels. The interpolated image has a pleasant and natural-looking result. In Fig. 3.4(c), no discontinuity can be observed from the 3D plot of the face portion of the interpolated image.

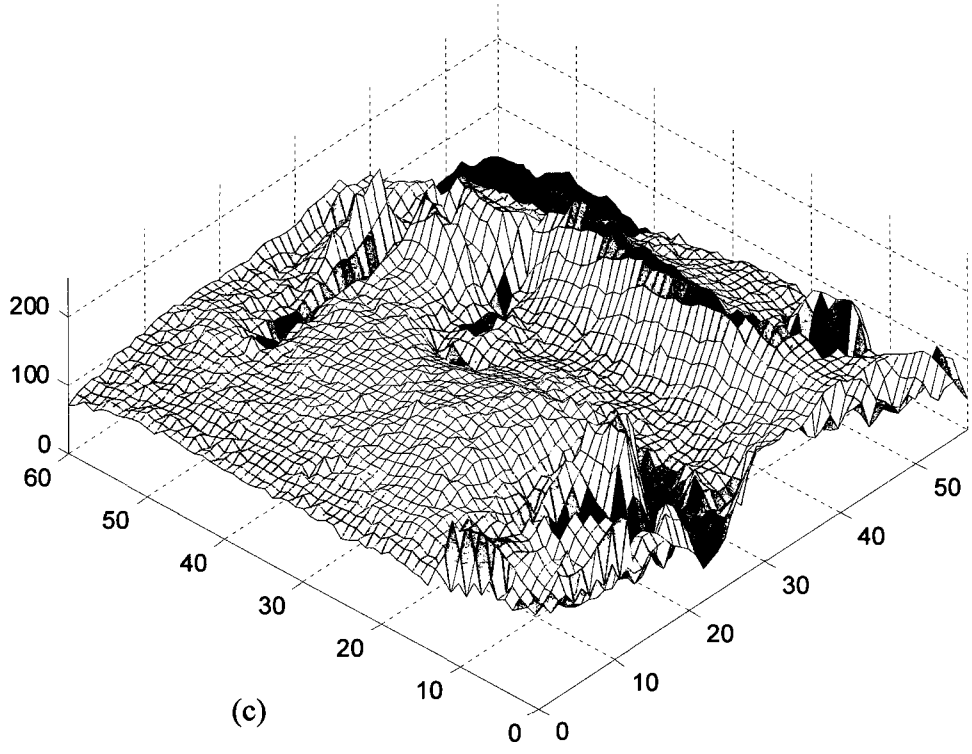
3.3 Summary

In this chapter, a new image interpolation model has been developed to meet the challenges facing both the spatial resolution enhancement for images and temporal resolution enhancement for video sequences.

This model is based on the finite element method technique, wherein the given image is segmented into a number of non-overlapping triangular elements. Each element is expressed as a function characterized by the values at its nodes. Then, with all the different elements assembled together and the boundary conditions applied, it has been shown that the unknown pixels can be interpolated by solving a set of linear equations. An iterative method has been used to obtain solutions to such a set of linear equations, instead of solving them directly. Simulations have been carried out to study the effect of the image block size and that of the initialization values on the interpolation results.



(a) (b)



(c)

Figure 3.4 : (a) Interpolated image resulting from the application of the generalized image interpolation model. (b) A sub-image from the face portion of the image in (a). (c) The 3D surface plot of the sub-image in (b)

Simulation results have shown that larger block size leads to better interpolation qualities at the cost of a higher computational complexity. They have also demonstrated that the initialization value does not affect the quality the interpolated image, while initialization with the mean value of the known pixels results in the smallest number of basic operations, and hence in the fastest convergence. By applying this model to a test image *Lena*, manipulated with arbitrarily shaped holes, the missing pixels have been interpolated. It has been shown that the interpolated pixels fit smoothly with the neighboring known and/or unknown pixels, and the interpolated image demonstrates a natural-looking appearance, with its 3D plot maintaining a continuity among the interpolated and known pixels.

In the following chapters, this image interpolation model is applied to the problems of spatial resolution enhancement of images and temporal resolution enhancement of encoded video sequences.

Chapter 4

Edge-preserving Iterative Refinement Interpolation for Spatial Resolution Enhancement of Images

4.1 Introduction

Based on the generalized image interpolation model developed in Chapter 3, an iterative refinement interpolation for spatial resolution enhancement of images will be proposed in this chapter [49]. The generalized image interpolation model will be also utilized and extended to support edge-preserving spatial resolution enhancement of images.

As already defined in Section 2.1, spatial resolution enhancement of images is a process of obtaining a high resolution image $H(m,n),(m,n) \in D^u$ by up-scaling a low resolution image $L(i,j)$, $(i,j) \in D$. At $(ui,\ uj)$, $H(m,n)$ has known pixels values, which are equal to $L(i,j)$, that is

$$H(ui,uj) = L(i,j) \tag{4.1}$$

This process can be better understood using Fig. 4.1, where we give an example for the case when the magnification factor $u = 2$. Our task is to interpolate the unknown pixels

at $(m,n)\,(m \neq ui \text{ or } n \neq uj)$, represented by white dots, based on the known pixels $H(ui,\ uj)$, represented by black dots.

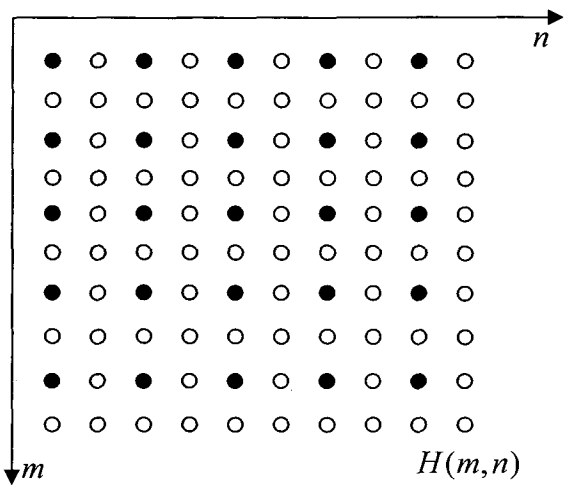


Figure 4.1: An example of the image up-scaling process. A 5×5 image is up-scaled to an image with 10×10, with $u = 2$. Black dots represent pixels from low resolution images $H(ui,uj) = L(i,j)$, and white dots representing the pixels to be interpolated, $H(m,n),(m \neq ui \text{ or } n \neq uj)$

4.2 Iterative Refinement Interpolation for Spatial Resolution Enhancement

In order to apply the FEM-based image interpolation model of Chapter 3, we first construct a continuous luminance function $H(x,y)$ in the domain Φ, which is defined as follows.

$$\Phi = [0, uW - 1] \times [0, uH - 1] \tag{4.2}$$

In this domain, we perform triangular segmentation and partition Φ into φ ($\varphi > 1$, φ an integer) non-overlapping triangular elements e_ν ($\nu = 1, \cdots, \varphi$), the nodes of which

correspond to the pixel locations in $H(m,n)$, as shown in Fig. 4.2(a). These elements are

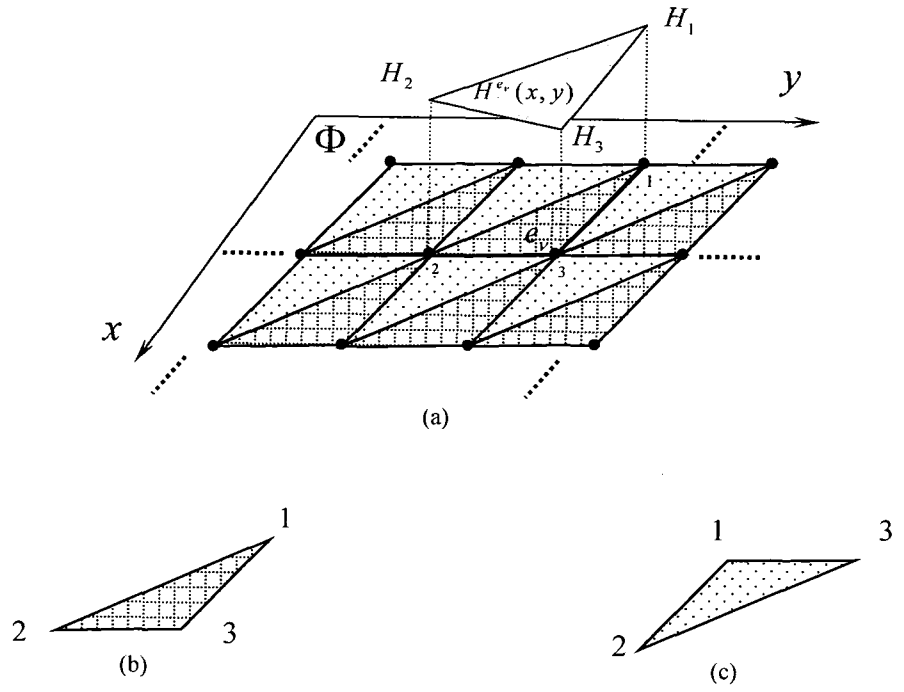


Figure 4.2: (a) Triangular elements partitioned in the domain Φ of $H(x,y)$. $H^{e_v}(x,y)$ represents the continuous function for the element e_v; (b) Type I local triangular element; (c) Type II local triangular element

classified into two types, type I and type II triangular elements, as shown in Fig. 4.2(b) and Fig. 4.2(c) respectively. The node at (ui, uj) corresponds to the known pixel with the value $L(i,j)$. Such nodes are referred to as the known nodes, and the remaining nodes as unknown nodes at (m,n), $(m \neq ui \text{ or } n \neq uj)$. Let $\mathrm{K} \subset D^u$ denote the set of known nodes and $U \subset D^u$ the set of unknown nodes; $U \bigcup \mathrm{K} = D^u$. Under the condition

$$H(x,y) = L(x,y), \ (x,y) \in \mathrm{K} \tag{4.3}$$

we have

$$H(x,y) = H^{e_v}(x,y) \quad (x,y) \in e_v \tag{4.4}$$

where

$$H^{e_v}(x,y) = \begin{pmatrix} H_1 & H_2 & H_3 \end{pmatrix} \begin{pmatrix} \omega_1(x,y) \\ \omega_2(x,y) \\ \omega_3(x,y) \end{pmatrix} \tag{4.5}$$

As illustrated in Fig. 4.2(a), $H^{e_v}(x,y)$, $(x,y) \in e_v$ is a continuous function with $H_i = H(x_i, y_i)$, $(i = 1, 2, 3)$, and $(x_i, y_i) \in D^u$, where (x_i, y_i) is the location of the *ith* node in the triangular element e_v. The shape functions are given by

$$\omega_1(x,y) = \frac{1}{2s}\begin{vmatrix} 1 & x & y \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{vmatrix}$$

$$\omega_2(x,y) = \frac{1}{2s}\begin{vmatrix} 1 & x_1 & y_1 \\ 1 & x & y \\ 1 & x_3 & y_3 \end{vmatrix} \tag{4.6}$$

$$\omega_3(x,y) = \frac{1}{2s}\begin{vmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x & y \end{vmatrix}$$

where s is the area of the triangular element e_v.

In most of the natural images, the change from one pixel value to that of an adjacent pixel is not abrupt. An interpolated pixel should fit smoothly with its immediate-neighboring pixels, which may be known pixels or themselves unknown pixels to be interpolated. To meet this requirement, we apply the optimization constraint that the surface represented by $H(x,y)$ has the smallest area. Thus, we need to find the pixel values for the unknown nodes to minimize the function

$$\iint_{\Phi}\sqrt{1+\left(\frac{\partial H(x,y)}{\partial x}\right)^2+\left(\frac{\partial H(x,y)}{\partial y}\right)^2}\,dxdy \tag{4.7}$$

In view of (4.4), the above problem is equivalent to finding the pixel values of $H(m,n)$ at each node $(m,n)\in U$ that minimizes the function

$$\sum_{v=1}^{\varphi}\iint_{e_v}\left[\left(\frac{\partial H^{e_v}(x,y)}{\partial x}\right)^2+\left(\frac{\partial H^{e_v}(x,y)}{\partial y}\right)^2\right]dxdy \tag{4.8}$$

We now recall that (4.8) has the same form as (3.6), and hence, it can be shown that the optimal value of an unknown node at $(m,n)\in U$ satisfies the equation

$$H(m,n)=C_V(m,n)+C_H(m,n) \tag{4.9}$$

with

$$C_V(m,n)=\begin{cases}\frac{1}{2}H(m+1,n) & if\ m=0\\ \frac{1}{2}H(m-1,n) & if\ m=uH-1\\ \frac{1}{4}\left[H(m-1,n)+H(m+1,n)\right] & if\ m\neq 0\ or\ uH-1\end{cases} \tag{4.10}$$

$$C_H(m,n)=\begin{cases}\frac{1}{2}H(m,n+1) & if\ n=0\\ \frac{1}{2}H(m,n-1) & if\ n=uW-1\\ \frac{1}{4}\left[H(m,n+1)+H(m,n-1)\right] & if\ n\neq 0\ or\ uW-1\end{cases} \tag{4.11}$$

Equation (4.9) is applicable to any node at $(m,n)\in U$. Therefore, determining the pixel value for each unknown node $(m,n)\in U$ requires solving a set of linear equations.

However, as pointed out in Section 3.2.3, the number of such equations being equal to the total number of unknown nodes in D^u could be very large. Solving these equations directly is, therefore, not efficient. Hence, as was done in solving (3.35)-(3.37), we employ the following iterative refinement procedure in order to obtain the pixel value at an unknown node:

$$H^{(k+1)}(m,n) = C_V^{(k)}(m,n) + C_H^{(k)}(m,n) \tag{4.12}$$

with

$$C_V^{(k)}(m,n) = \begin{cases} \frac{1}{2}H^{(k)}(m+1,n) & \text{if } m = 0 \\ \frac{1}{2}H^{(k+1)}(m-1,n) & \text{if } m = uH-1 \quad (k \geq 0) \\ \frac{1}{4}\left[H^{(k+1)}(m-1,n) + H^{(k)}(m+1,n)\right] & \text{if } m \neq 0 \text{ or } uH-1 \end{cases} \tag{4.13}$$

$$C_H^{(k)}(m,n) = \begin{cases} \frac{1}{2}H^{(k)}(m,n+1) & \text{if } n = 0 \\ \frac{1}{2}H^{(k+1)}(m,n-1) & \text{if } n = uW-1 \quad (k \geq 0) \\ \frac{1}{4}\left[H^{(k+1)}(m,n-1) + H^{(k)}(m,n+1)\right] & \text{if } n \neq 0 \text{ or } uW-1 \end{cases} \tag{4.14}$$

Obviously, the above procedure is not applied to the known nodes involved in (4.12), with $H^{(k+1)}(x,y) = H^{(k)}(x,y), (x,y) \in \mathrm{K}$. When the stop criterion is met, we obtain the pixel value for this unknown node. It can be seen that this iterative scheme is independent of the magnification factor u. Hence, u can be any positive integer, which is not the case with any of the other existing methods. We will discuss this point further in Section 4.3, where we compare the proposed scheme with other existing interpolation methods.

Based on the above discussion, we now formulate an algorithm for the interpolation of the pixel values of the unknown nodes at $(m,n) \in U$. Let $\vec{H}$ be a vector, each component of which corresponds to a value of a pixel at an unknown node $(m,n) \in U$. Let $H^{(0)}(m,n)$ be the initial value of such a pixel, and be given by

$$H^{(0)}(m,n) = L(floor(m/u), floor(n/u)) \quad (m,n) \in U \tag{4.15}$$

Let $MAPD(\cdot,\cdot)$ represent an operator that calculates the maximum of the absolute difference of the pixel values and ε a pre-specified tolerance. The operation of $MAPD(\cdot,\cdot)$ is restricted to integer accuracy so that we can set the tolerance $\varepsilon = 0$. This procedure starts with the initialization of $\vec{H}^{(0)}$ with $H^{(0)}(m,n)$, and $k = 0$. For each node at $(m,n) \in U$, (4.12) is used to calculate $H^{(k+1)}(m,n)$. If $MAPD(\vec{H}^{(k+1)}, \vec{H}^{(k)}) > 0$, then k is incremented by unity, and $H^{(k+1)}(m,n)$ is computed again for each node at $(m,n) \in U$. This process is repeated until the value of $MAPD(\vec{H}^{(k+1)}, \vec{H}^{(k)})$ becomes zero. Then, the value of $\vec{H}^{(k+1)}$ determined at the last step is assigned to the vector $\vec{H}$. The vector $\vec{H}$ obtained from the above algorithm contains all the interpolated pixel values for the unknown nodes in D^u, and the up-scaled image $H(m,n)$ is thus obtained. The flow chart of the procedure is shown in Fig. 4.3.

Since the above scheme can be applied to any sub-image block of $H(m,n)$, we partition $H(m,n)$ into a number of sub-image blocks, so that the computational effort for interpolation gets distributed among these blocks. At the end of this iterative refinement process, the interpolated pixels will exhibit a smooth variation with the known pixels $H(ui, uj)$, and the neighboring interpolated pixels as well. In the next section, we

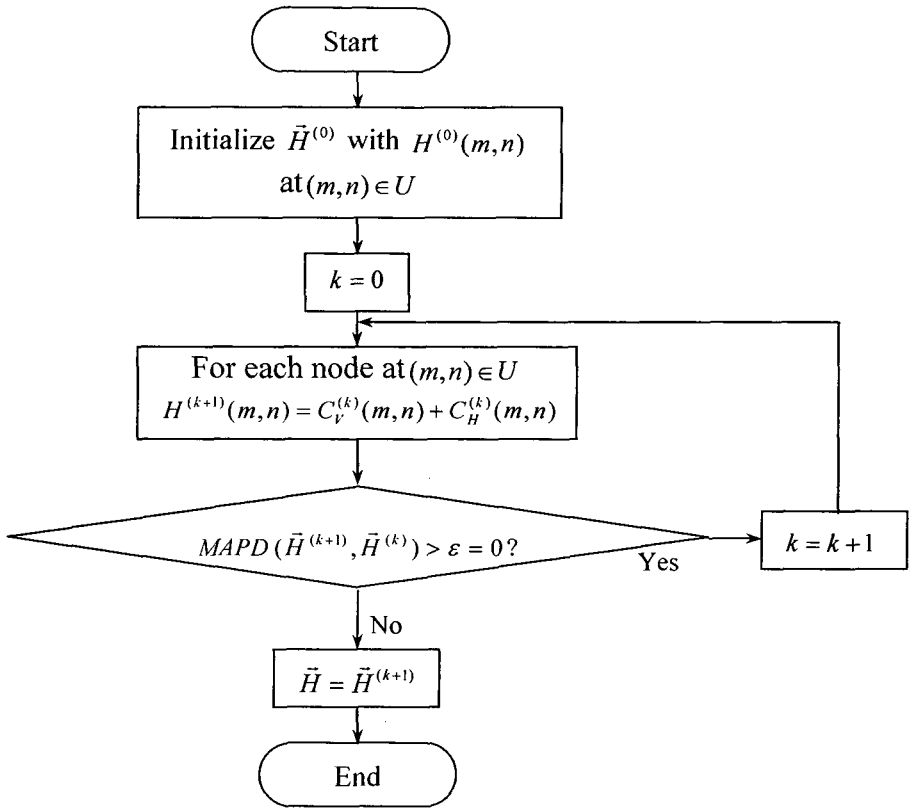


Figure 4.3: Flow chart of the iterative refinement interpolation scheme.

extend this proposed technique to develop an edge-directed iterative refinement interpolation scheme for spatial resolution enhancement of images.

4.3 Edge-preserving Iterative Refinement Interpolation for Spatial Resolution Enhancement

As discussed in Chapter 2, the edges in a natural image constitute one of the most prevalent features for human visual systems. Variations in pixel values always occur, where a dominant edge is present. Along the edge direction, the pixel values change more slowly than they do across an edge. To address this problem, we apply an edge-

preserving refinement phase to the up-scaled image $H(m,n)$, obtained from the previous iterative refinement process. We first detect the orientation of the dominant sharp edge in the 5×5 image block BLK in the original low resolution image $L(i,j)$, as shown in Fig. 4.4(a). Based on the detected orientation, we apply the equation (4.9) to perform an edge-preserving refinement phase to the unknown pixels in the corresponding image BLK^u (a block with a size of $5u\times5u$) in $H(m,n)$. Fig. 4.4(b) shows BLK^u when u=2.

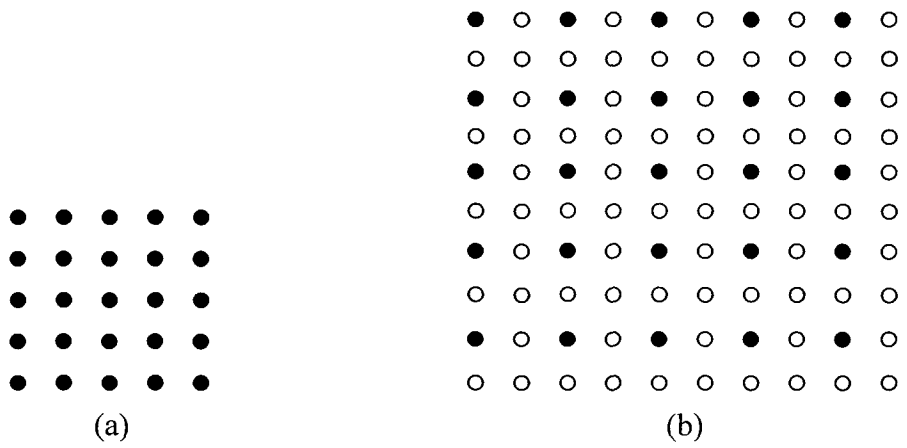


Figure 4.4: (a) A 5×5 image block BLK in the original low resolution image $L(i,j)$. (b) The corresponding image block BLK^u of the up-scaled image $H(m,n)$ with $u=2$, the black dots representing the pixels from $L(i,j)$, and white dots the pixels to be interpolated.

In order to detect the dominant sharp edge in BLK, we need to use a gradient estimator [65]. Some estimators, such as the Sobel operator or the Prewitt operator, can provide good edge detection, but are computationally expensive. On the other hand, the Roberts operator uses a small mask, and is computationally less expensive, but produces

weak responses to genuine edges unless they are very sharp. However, since in our case, we are interested in detecting only the dominant sharp edges in BLK, we select the Roberts operator as the gradient estimator. As will be seen from simulation results in Section 4.4, Roberts operator accomplishes this task very well. Through the use of the Roberts operator, the angular direction and magnitude the gradient of a pixel at (i, j) in the block BLK are determined. For each pixel at (i, j), we first obtain

$$g_x = L(i, j) - L(i+1, j+1) \tag{4.16}$$

$$g_y = L(i, j+1) - L(i+1, j) \tag{4.17}$$

The magnitude $G(i, j)$ and the angular direction $\theta(i, j)$ of the gradient at (i, j) are given by

$$G(i, j) = \sqrt{g_x^2 + g_y^2} \tag{4.18}$$

and

$$\theta(i, j) = \tan^{-1}(g_y / g_x) \tag{4.19}$$

To reduce the computational effort, the gradient magnitude is approximated as the sum of the absolute values of g_x and g_y [65], that is

$$G(i, j) = |g_x| + |g_y| \tag{4.20}$$

To carry out the directional classification of the pixels, we use an approach similar to that in [53]. Consequently, the angular direction is rounded to the nearest multiple of 22.5° by

$$\theta_{round}(i, j) = \left[\theta(i, j) / 22.5^\circ\right]_{round} \cdot 22.5^\circ \tag{4.21}$$

Thus, $\theta_{round}(i,j)$ corresponds to one of the 8 orientations that are equally spaced from 0° to 157.5°, and denoted as ψ_r $(r = 0,1,...,7)$

$$\psi_r = r \cdot 22.5^\circ \ (r = 0,1,...,7) \tag{4.22}$$

In order to detect the dominant edge that may exist in the block, we compare $G(i,j)$ to a threshold Th, which is locally adapted for a low resolution image $L(i,j)$ [65],

$$Th = \rho \cdot m_G \qquad (\rho > 0) \tag{4.23}$$

where m_G is the arithmetic mean of the gradient values of all the pixels in $L(i,j)$. If the amplitude of the gradient of a pixel is greater than Th, we designate it to be an edge pixel. Each edge pixel is then taken into consideration in order to find the dominant edge orientation in BLK. Eight registers, R_0 through R_7, are associated for the different orientations defined in (4.22). If $\theta_{round}(i,j) = \psi_r$, we increment the value of R_r $(r = 0,1,...,7)$ by $G(i,j)$. After all of the edge pixels in the block BLK are taken into consideration in this counting process, the register with the largest value, R_{Max}, leads to an edge orientation ψ_D, which usually represents the orientation for the dominant edge. However, it is possible that more than one edge orientation is detected in BLK, and the values of the corresponding registers are quite comparable. This suggests that BLK contains more than one discernible edge, and an interpolation method along only a given orientation is not desired. In view of this, the following criterion is employed.

For any $R_r \neq R_{\max}$, if $R_r > w \cdot R_{\max}$ (w is set to 0.2), then the gradient estimator determines that BLK has no dominant edge and therefore, no further action is needed.

Otherwise, ψ_D is considered as the orientation for a dominant edge. Then, the edge-directed interpolation should be carried out for the unknown pixels in BLK^u along ψ_D.

If $\psi_D = 0^\circ$ or $\psi_D = 90^\circ$, the interpolation for an unknown pixel in BLK^u is carried out in the vertical or horizontal direction, respectively. Based on (4.9), the optimal value for an unknown pixel is given by

$$\begin{cases} H(m,n) = 2 \cdot C_V(m,n), & \psi_D = 0^\circ \\ H(m,n) = 2 \cdot C_H(m,n), & \psi_D = 90^\circ \end{cases} \tag{4.24}$$

Let us now consider the case when ψ_D is neither 0° nor 90°. Let the coordinates of an unknown pixel A be (m,n) in the original coordinate system (marked as dotted lines in Fig. 4.5), and (m',n') in the new coordinate system, which is obtained by rotating

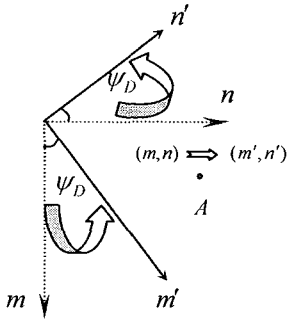


Figure 4.5: An unknown pixel A in the two coordinate system.

counter-clockwise the original coordinate system by ψ_D. An immediate neighbor of pixel A in the new coordinate system can be denoted as $(m'+l, n'+q)$, $(l = -1, 0, 1; q = -1, 0, 1; l \neq q)$. Its counterpart in the original coordinate system can be obtained from the following equation:

$$\begin{pmatrix} m'+l \\ n'+q \end{pmatrix} = \begin{pmatrix} m \\ n \end{pmatrix} + \begin{pmatrix} l & -q \\ q & l \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} \tag{4.25}$$

where

$$\Delta x = \arg\min_{h} \{ [h \cdot \tan(\psi_D)]_{round} > 0 \} \quad h \text{ an integer} \tag{4.26}$$

$$\Delta y = [\Delta x \cdot \tan(\psi_D)]_{round} \tag{4.27}$$

Therefore, the optimal value for an unknown pixel A at (m,n) is derived from (4.10) as

$$H(m',n') = 2 \cdot C_V(m',n') \quad \psi_D \neq 0^\circ,\ 90^\circ \tag{4.28}$$

Equations (4.24) and (4.28) are utilized to obtain the pixel value at the unknown node $(m,n) \in U$ in an iterative manner, in the same way as (4.12) is used in obtaining $H(m,n)$. The edge-preserving refinement phase is performed in the orientation ψ_D of the detected dominant edge, thus maintaining the smooth variation along that direction. When no dominant edge is detected in a BLK, then no edge-preserving refinement is performed in the BLK^u; in such a case, the value for each pixel in BLK^u remains unchanged. As will be shown later in Section 4.4, this process properly preserves the dominant edge, thus improving the quality of the up-scaled image.

4.4 Simulation Results

To test the performance of edge-preserving iterative refinement interpolation (EPIR) scheme developed in Section 4.3 for image up-scaling, we select four interpolation methods for comparison. Three of these, namely, the nearest-neighbor, bilinear and

bicubic are conventional interpolation methods [1]-[3], and the fourth is the edge-directed interpolation method (EDIM) proposed in [18]. Although quite a few edge-directed interpolation methods have been proposed, the EDIM method is considered as one of the best schemes for up-scaling images with fine textures and edges. Four natural test images of size 512×512, *Lena*, *Airplane*, *Sailboat*, and *Baboon*, are used as the test images. In this experiment, we model the low resolution image as a down-scaled version of an original test image, by a factor of u .The four interpolation methods chosen for comparison, along with the proposed EPIR scheme, are used to up-scale the low resolution image by the magnification factor u. We recall that our proposed scheme consists of applying the FEM-based iterative refinement interpolation to obtain the up-scaled image $H(m,n)$, followed by the edge-preserving refinement process. A block size of $8u \times 8u$ is chosen as the interpolation block size for the FEM-based iterative refinement, and the edge preserving refinement phase is carried out with a block size of $5u \times 5u$. The threshold value *Th* given by (4.23) is chosen to be $3m_G$ in the detection of an edge pixel.

We use PSNR as a metric to evaluate the objective quality of the up-scaled images obtained by using the five interpolation methods mentioned above. The PSNR values obtained for the four different test images for a magnification factor of two are presented in Table 4.1. It is seen from this table that for all the test images, the EDIM yields better results than the three conventional interpolation methods do; however, the proposed EPIR scheme provides the best performance.

In order to bring out further the advantage of the proposed scheme, a visual comparison is made to evaluate the quality of the up-scaled images. Results using the

various interpolation schemes on the test images *Lena*, *Airplane*, *Sailboat*, and *Baboon* are presented in Figs. 4.6-4.9, respectively, wherein we present only certain portions of the interpolated frames that contain sharp edges and fine textures. This is done so in order to highlight the difference in the quality amongst the interpolated frames for each of the test images.

Table 4.1: PSNR (dB) results of the up-scaled images using various schemes

Test image	Nearest	Bilinear	Bi-cubic	EDIM	Proposed
Lena	27.92	29.48	32.66	33.56	34.48
Airplane (F-16)	25.77	27.41	27.42	27.53	27.80
Sailboat	23.69	25.16	25.34	25.64	26.04
Baboon	20.16	21.54	22.80	23.29	23.76

It is seen from Fig. 4.6(b) that the nearest-neighbor interpolation method exhibits discernible pixelization artifacts. Bilinear interpolation (see Fig. 4.6(c)) presents smooth results, but the up-scaled image appears blurred in areas where there should have been fine texture and strong edges, such as in the face area and the rim of the hat. Bicubic interpolation (Fig. 4.6(d)) produces sharper results in these areas compared to the bilinear method, but with increased jaggy effects along the rim of the hat. As seen from Fig. 4.6(e), the EDIM produces a visually pleasant result with fine texture and edges properly reconstructed. However, some small fleck-like interpolation artifacts along the hat rim can be observed. Our proposed scheme produces visually pleasant results without

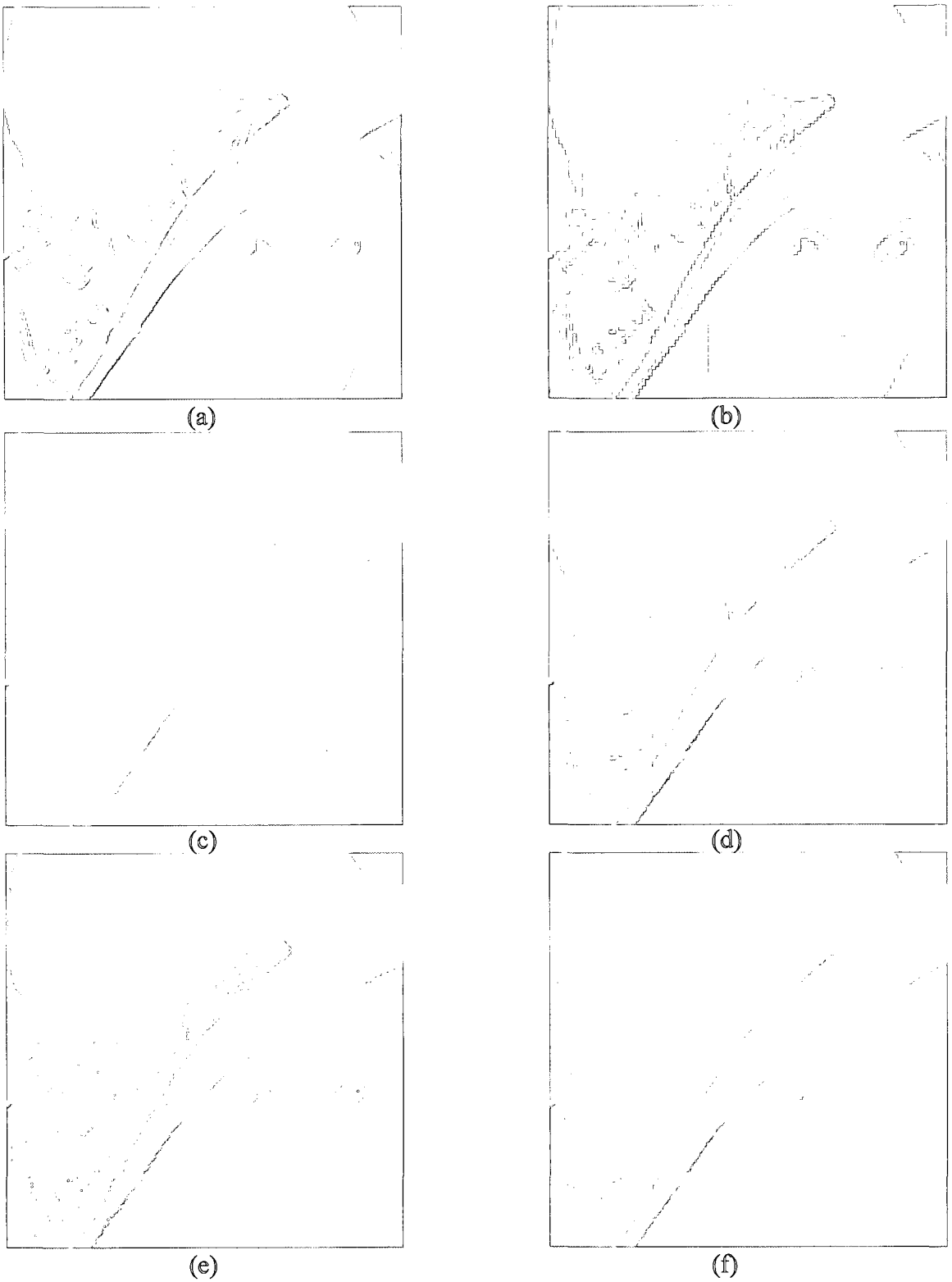


Figure 4.6: Visual comparison of different interpolation schemes on test image *Lena*. (a) Portion of original image, portions of interpolated image by the (b) nearest interpolation, (c) bilinear interpolation, (d) bicubic interpolation, (e) EDIM, and (f) our proposed EPIR scheme

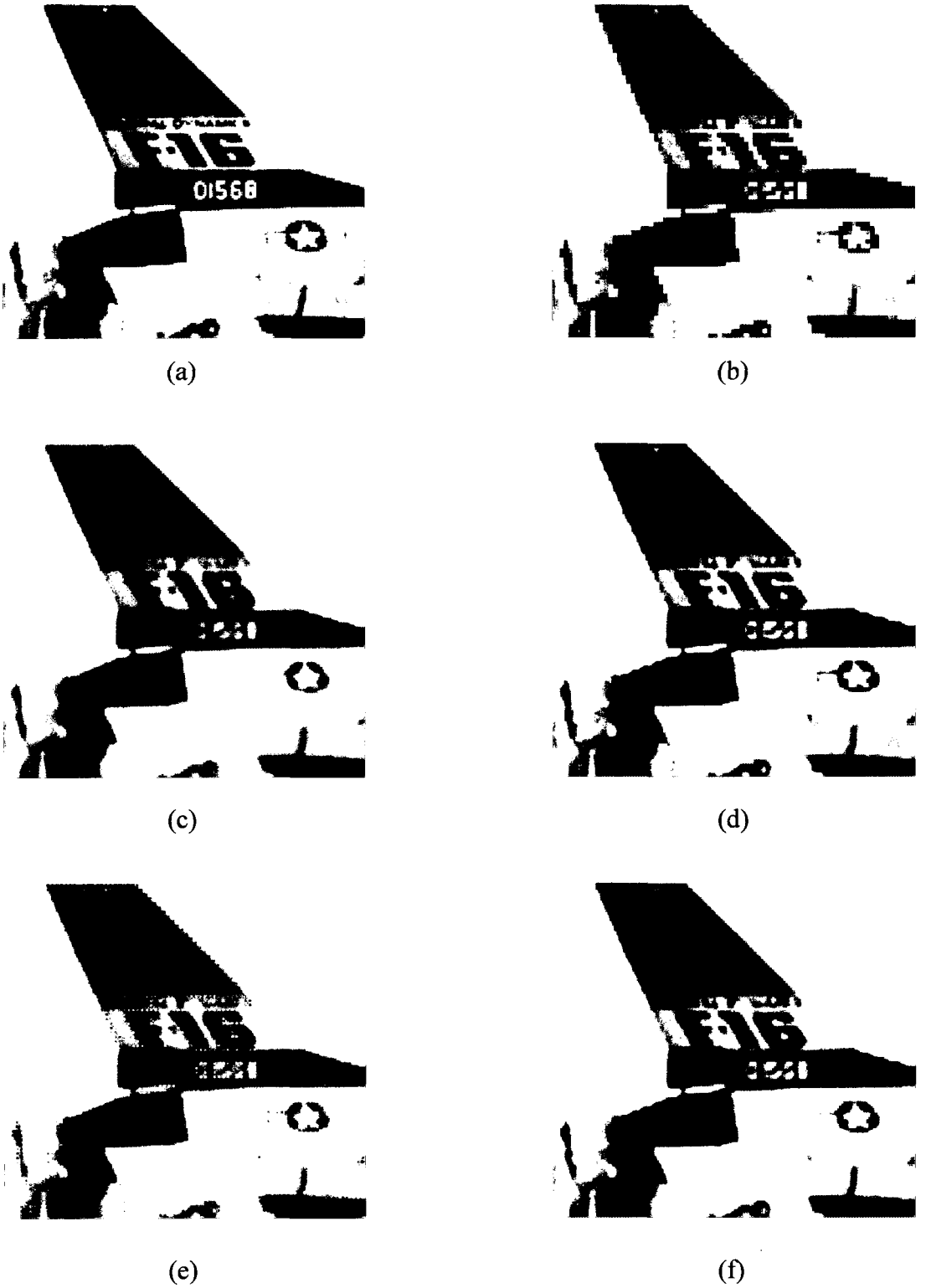


Figure 4.7: Visual comparison of different interpolation schemes on test image *Airplane*. (a) Portion of original image, portions of interpolated image by the (b) nearest interpolation, (c) bilinear interpolation, (d) bicubic interpolation, (e) EDIM, and (f) our proposed EPIR scheme

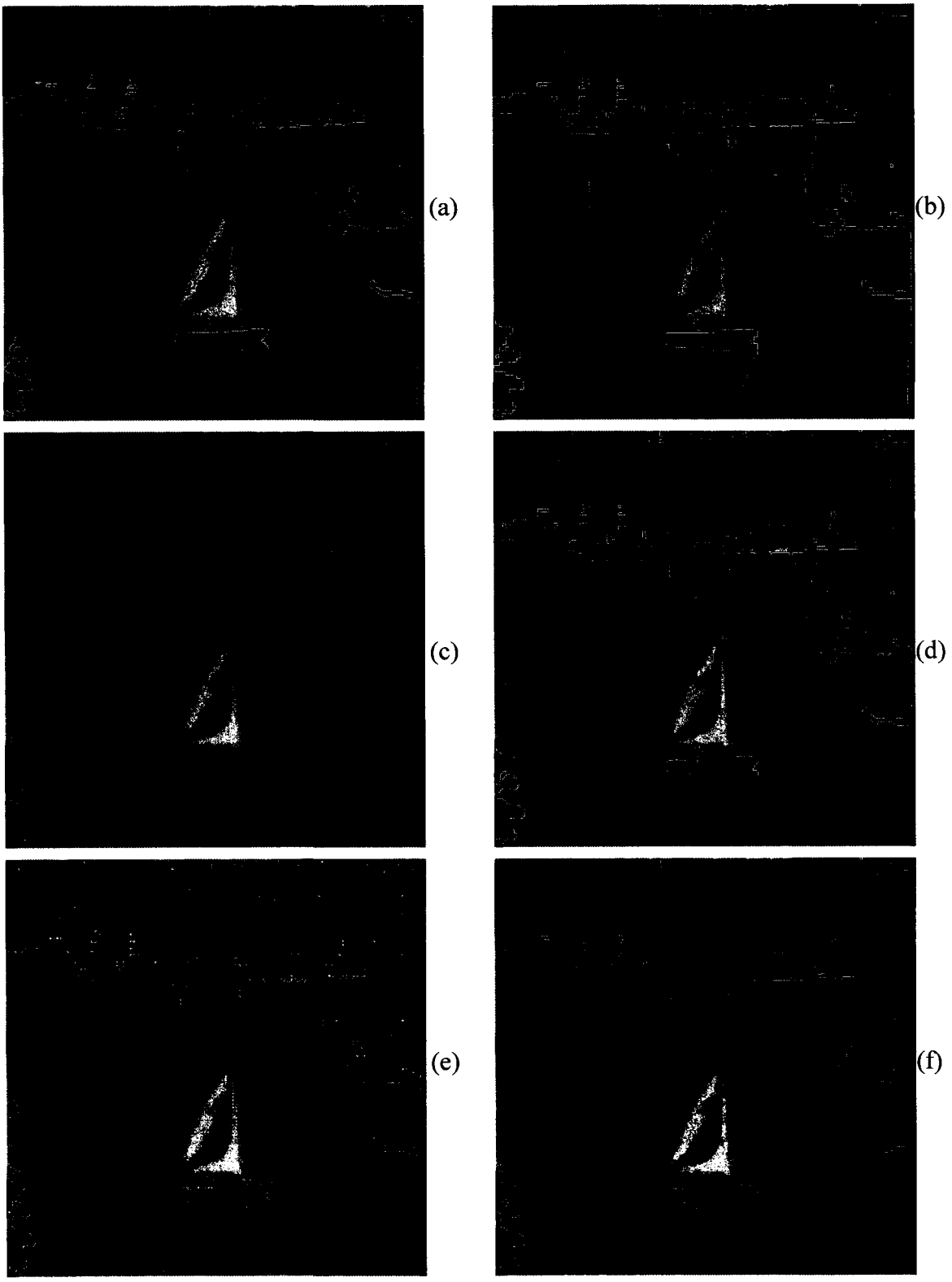


Figure 4.8: Visual comparison of different interpolation schemes on test image *Sailboat*. (a) Portion of original image, portions of interpolated image by the (b) nearest interpolation, (c) bilinear interpolation, (d) bicubic interpolation, (e) EDIM, and (f) our proposed EPIR scheme

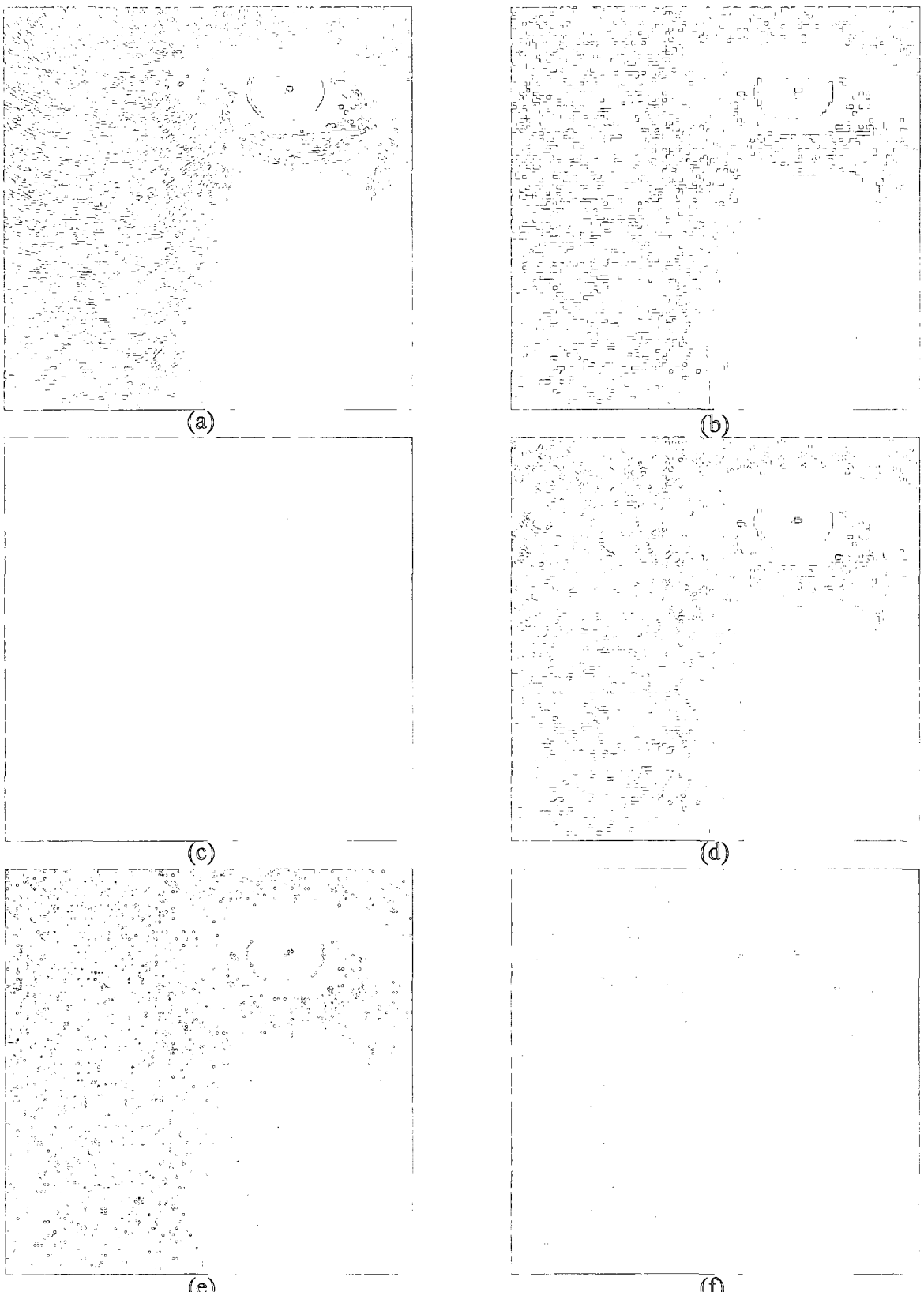


Figure 4.9: Visual comparison of different interpolation schemes on test image *Baboon*. (a) Portion of original image, portions of interpolated image by the (b) nearest interpolation, (c) bilinear interpolation, (d) bicubic interpolation, (e) EDIM, and (f) our proposed EPIR scheme

displaying any artifacts such as pixelization, blurred effects, jaggy edges, and also presents fine texture and edge details in the up-scaled image, as seen from Fig. 4.6(f).

Similar results can be observed from Figs. 4.7-4.9 for the other test sequences, *Airplane*, *Sailboat*, and *Baboon*, respectively. As expected, the nearest-neighbor interpolation method generates pixelization effects, and bilinear interpolation method tends to blur the up-scaled images although the images look smooth. Bicubic interpolation provides jaggy effects along the edges. The EDIM method presents visually pleasant images, but it can be noticed that some small artifacts are present along the edges. In contrast, our scheme presents sharper edges and finer textures. This is due to the fact that, in order to interpolate an unknown pixel, our interpolation scheme not only exploits neighboring known pixels, but also takes into account the neighboring unknown pixels through an iterative refinement technique. This provides a spatial continuity between the unknown pixels as well, and therefore helps to reconstruct more natural results in areas with fine textures. In the presence of a strong dominant edge in a sub-image block to be interpolated, the edge-preserving iterative refinement process is performed along the detected dominant edge, and thus the smooth variation along that direction is maintained.

As for the computational complexity, the proposed EPIR scheme requires very much less basic operations than the EDIM does. Table 4.2 shows the total number of basic operations needed by using these two schemes to interpolate the various test images with $u=2$.

Table 4.2: Computational complexity (in number of basic operations) comparison between EDIM and the proposed EPIR ($u = 2$)

Methods	*Lena*	*Airplane (F-16)*	*Sailboat*	*Baboon*
EDIM	35,586,048	34,904,474	28,770,304	55,625,408
Proposed	5,714,739	5,400,166	5,223,219	6,225,920

Another advantage of our scheme is reflected by its easy implementation. Since in the existing interpolation schemes, the unknown pixels are interpolated based only on the estimation made on the known pixels, these schemes require more than one step to complete the interpolation process even when $u = 2$. Obviously, the number of steps increases with the increasing value of the magnification factor u. Moreover, some of the proposed schemes have been formulated only when the magnification factor u is two [18], [21]; hence, such a scheme needs to be applied n times to up-scale an image by a magnification factor of $u = 2^n$ ($n > 1$ and an integer). Further, when the magnification factor u is not an integer power of 2 (for example, $u = 3$ or 6), neither [18] nor [21] can be employed directly. To deal with such a case, the authors in [21] have suggested the use of a bilinear or bicubic interpolation method as an additional step for interpolation after up-scaling by a factor of 2^n using their method. For example, in order to up-scale an image by a magnification factor $u = 3$, the authors first apply their method for $u = 2$, and then use bicubic interpolation method to scale the image by a factor of 1.5 to achieve an overall magnification factor of $u = 3$. Even though the authors in [18] have not suggested a method as to how to deal when u is not a power of two, one could adapt here also, the

procedure suggested in [21], namely the use of bilinear or bicubic method as an additional step. Obviously, in such a case the advantages associated with the schemes in [18] and [21] cannot be fully realized. In contrast, in our scheme, the algorithm needs to be applied only once to interpolate all the unknown pixels in the up-scaled image, irrespective of the value of the magnification factor u.

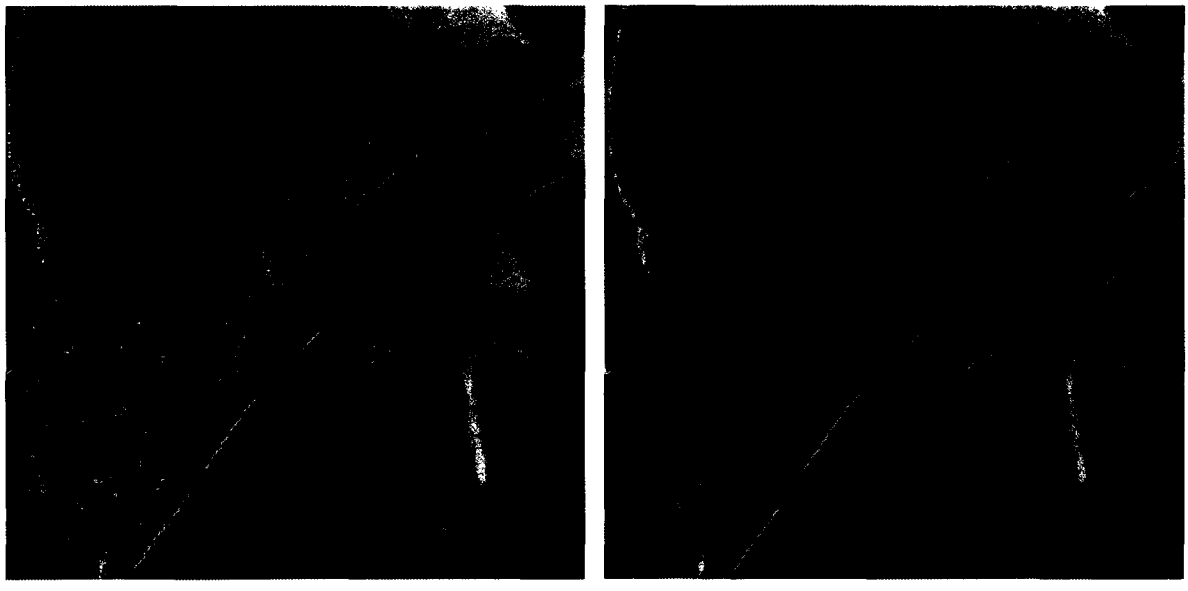
In order to compare the performance of the proposed EPIR scheme with that of the existing schemes for a magnification factor u that is not a power of two, we choose u to be three and the EDIM scheme of [18] along with the bicubic method suggested in [21]. The EDIM scheme has been chosen, since it has the best performance amongst all the existing methods. For this purpose, four natural images (512×512 pixels), *Lena*, *Airplane*, *Sailboat*, and *Baboon*, are used as the test images. The four test images are first cropped to the resolution of 510×510, and then down-sampled to 170×170 low resolution images. The EDIM scheme combined with the bicubic method (EDIM+bicubic) and the proposed EPIR scheme are implemented to obtain the 510×510 high resolution images. The PSNR is used as the metric to evaluate the objective quality of the up-scaled images. The PSNR values obtained for the four test images for a magnification factor of three are presented in Table 4.3. From this table, it is seen that the proposed EPIR scheme outperforms the EDIM+bicubic scheme.

A visual comparison is also made to evaluate the subjective quality of the up-scaled images, and the results presented in Figs. 4.10 - 4.13. It can be seen that the EDIM+bicubic and EPIR schemes produce images of about the same visual quality. The EDIM+bicubic scheme reveals some small fleck-like interpolation artifacts along the

edges, even though it appears to produce sharper edges. The proposed EPIR scheme produces visually pleasant results without displaying any such artifacts.

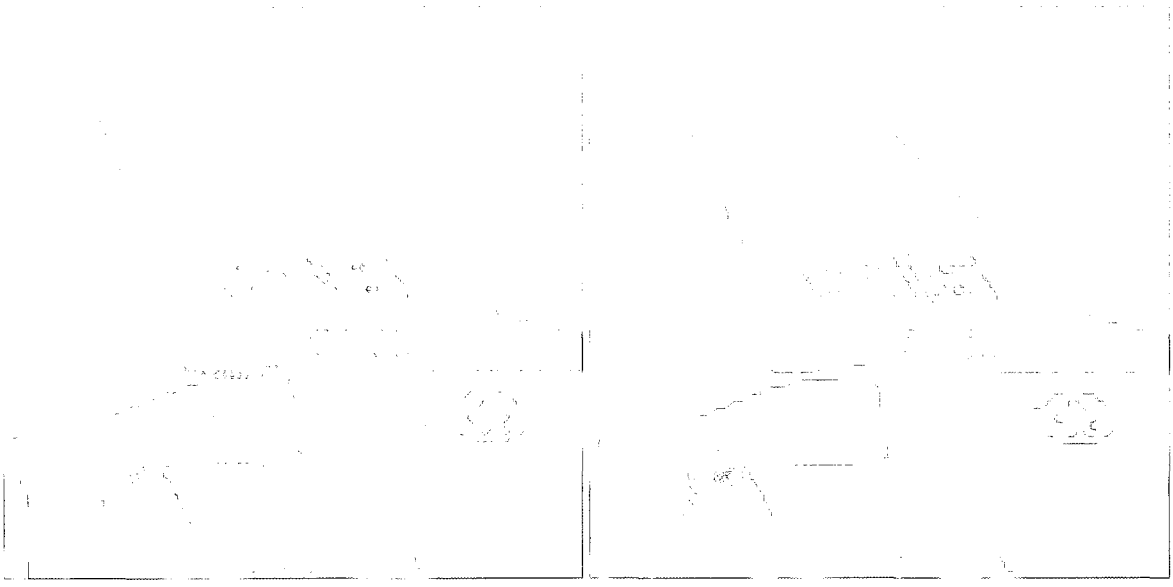
Table 4.3: PSNR (dB) results of the up-scaled images by a factor of three using the EDIM+bicubic and proposed EPIR schemes

Test image	EDIM+bicubic scheme	Proposed EPIR scheme
Lena	30.82	31.38
Airplane (F-16)	25.10	25.28
Sailboat	23.33	23.96
Baboon	20.54	20.76



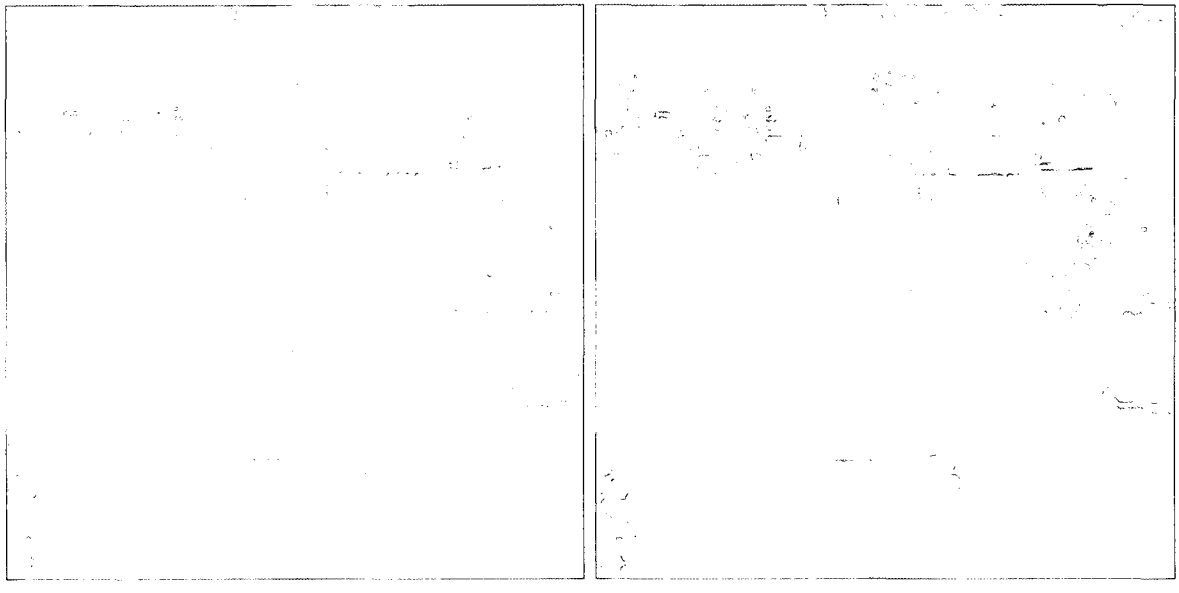
(a) (b)

Figure 4.10: Portions of test Image *Lena* up-scaled by $u = 3$ using (a) the EDIM+bicubic scheme, (b) our proposed EPIR scheme



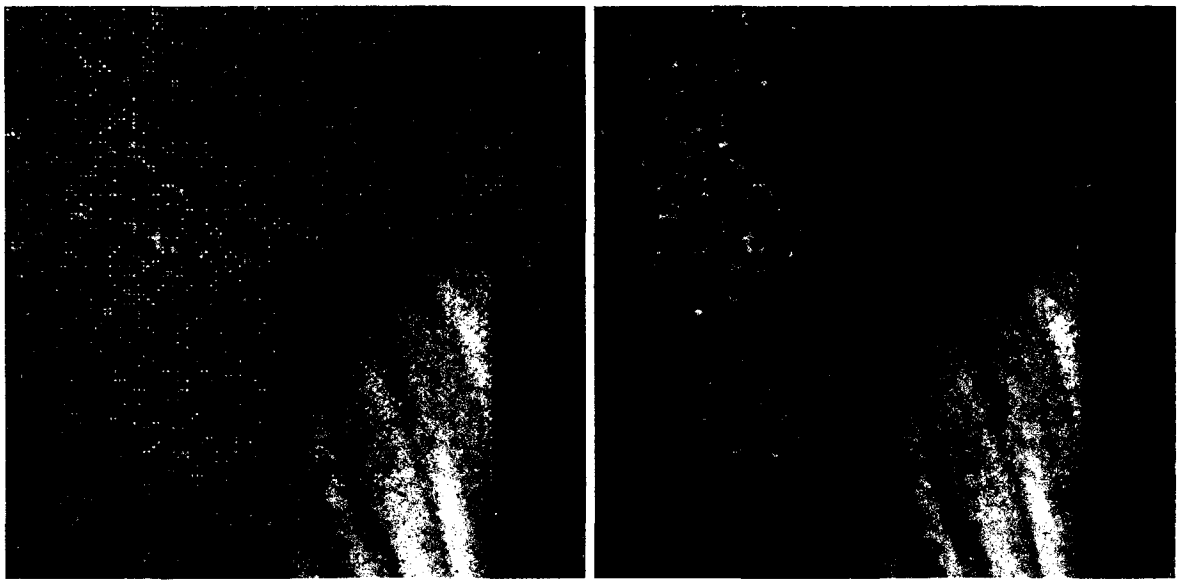
(a) (b)

Figure 4.11: Portions of test Image *Airplane* up-scaled by $u = 3$ using (a) the EDIM+bicubic scheme, (b) our proposed EPIR scheme



(a) (b)

Figure 4.12: Portions of test Image *Sailboat* up-scaled by $u = 3$ using (a) the EDIM+bicubic scheme, (b) our proposed EPIR scheme



(a) (b)

Figure 4.13: Portions of test Image *Baboon* up-scaled by $u = 3$ using (a) the EDIM+bicubic scheme, (b) our proposed EPIR scheme

A comparison of computational complexity of these two methods is also made, and the results presented in Table 4.4, which shows the total number of basic operations needed to interpolate the various test images with $u = 3$. It can be seen that the proposed EPIR scheme requires very much less basic operations than the EDIM+bicubic does.

Table 4.4: Computational complexity (in number of basic operations) of the EDIM+bicubic and proposed EPIR schemes for u=3

Methods	*Lena*	*Airplane (F-16)*	*Sailboat*	*Baboon*
EDIM+bicubic scheme	16,559,700	15,681,140	12,976,100	22,744,300
Proposed EPIR scheme	6,334,880	5,964,960	5,756,880	6,936,000

4.5 Summary

In this chapter, a novel edge-preserving iterative refinement scheme, has been developed for image up-scaling using the generalized image interpolation model developed in Chapter 3. The interpolation scheme is not restricted to the neighboring pixels with known values; it also takes into account the neighboring pixels with unknown values through an iterative refinement technique, in order to provide a spatial continuity between the unknown pixels as well. Therefore, the proposed method is capable of reconstructing natural-looking images in areas with fine textures. The edge-preserving iterative refinement process provides a smooth variation along a dominant edge in the up-scaled image. Simulation results have shown that the proposed method results in up-scaled images with better subjective and objective qualities than that provided by the existing interpolation schemes, including the EDIM [18], which has been considered as one of the best methods in its category. Compared to the EDIM, the proposed method has a much lower computational complexity. Also, it is very general in that it is not only capable of up-scaling an image by an arbitrary magnification factor u that is not restricted to be an integer power of 2, but also needs to be applied only once irrespective of the integer value of the magnification factor. In contrast, the existing methods all require multiple steps, to be run several times, and may need the use of conventional interpolation methods (such as bicubic or bilinear interpolation) when the magnification factor is not an integer power of 2.

Chapter 5

Error Concealment-based MCI for Temporal Resolution Enhancement of Encoded Video Sequences

5.1 Introduction

As discussed in Chapter 2, our research study in the temporal resolution enhancement of encoded video sequences is focused on block-based MCI schemes. We recall that the task of block-based MCI is to interpolate $\hat{FRM}_{t-\alpha}$ for time $t-\alpha$, based on the successive reconstructed frames FRM_{t-1} and FRM_t, at time $t-1$ and t, respectively. As defined in Section 2.2, $\vec{p}$ denote the position vector of a pixel at point P in a frame, $F_t(\vec{p})$ represent the pixel intensity at P inside the frame FRM_t, and $\vec{V}_i$ refer to the transmitted motion vector of a block B_i in the frame FRM_t. Since $\vec{V}_i$ is assigned to any pixel at $\vec{p} \in B_i$ the motion vector between FRM_t and $\hat{FRM}_{t-\alpha}$ is equal to $\vec{V}_i^{\alpha}$. For a pixel at $P \in B_i$ in FRM_t, the corresponding pixel in $\hat{FRM}_{t-\alpha}$ is at P_α. Then, the pixel value at P_α in $\hat{FRM}_{t-\alpha}$ is given by $\hat{F}_{t-\alpha}(\vec{p}_\alpha)$, which is equal to $F_t(\vec{p})$. As explained in Section 2.2.1,

this motion compensation process can cause the occurrence of overlapped and unfilled pixels, due to the mechanism used for the block-based motion estimation and motion compensation. These overlapped and unfilled pixels comprise the interpolation errors in the interpolated frames. If these interpolation errors are not concealed properly, the interpolated frame will display annoying stripe effects, leading to an unsatisfactory subjective quality. In this chapter, we develop an error concealment-based MCI scheme to conceal the interpolation errors left by the motion compensation process [47], [48], [50].

5.2 Error Concealment-Based MCI Schemes

Unlike the existing block-based MCI schemes, which involve computationally expensive pixel classification to deal with the interpolation errors, our scheme is based on an error concealment technique. In order to develop a proper scheme to conceal the interpolation errors, we first make a study of the number of such errors that could occur in an interpolated frame. Eight original 30 frames/s test sequences are sub-sampled to 15 frames/s, by skipping every other frame in a number of different sequences including *Foreman*, *Suzie*, *Miss America*, *Mother & Daughter*, *Stefan*, *Coast Guard*, *Container*, and *News*. All the skipped frames are interpolated by motion compensation, which makes use of the sub-sampled test sequences and the motion vector field generated through the full search block matching algorithm applied to the sequences at 15 frames/s. The average number of interpolation errors in an interpolation frame for each of the sequences is calculated and presented in Table 5.1. Similar study is carried out using 10 frames/s

test sequences (i.e., 2 out of 3 original frames are skipped), and the result is given in Table 5.2.

Table 5.1: Average number of interpolation errors in an interpolated frame for various original test sequences (15 frames/s)

Sequences (15 frames/s)	Format	Image size	Number of interpolation errors	Percentage of interpolation errors (%)
Mother&Daughter	QCIF	25344	232	0.92%
Miss America	QCIF	25344	252	0.99%
Suzie	QCIF	25344	1041	4.11%
Foreman	QCIF	25344	1063	4.19%
Container	CIF	101376	312	0.31%
News	CIF	101376	992	0.98%
Coast Guard	CIF	101376	4118	4.06%
Stefan	CIF	101376	6388	6.30%

Table 5.2: Average number of interpolation errors in an interpolated frame for various original test sequences (10 frames/s)

Sequences (10 frames/s)	Format	Image size	Number of interpolation errors	Percentage of interpolation errors (%)
Mother&Daughter	QCIF	25344	243	0.96%
Miss America	QCIF	25344	264	1.04%
Suzie	QCIF	25344	1091	4.30%
Foreman	QCIF	25344	1114	4.40%
Container	CIF	101376	400	0.39%
News	CIF	101376	1120	1.10%
Coast Guard	CIF	101376	5233	5.16%
Stefan	CIF	101376	8053	7.94%

It can be seen from these tables that the average number of interpolation errors is related to how fast the motion is in a certain sequence. We find that the total number of pixels caused by interpolation errors is, on the average, only less than 5% of the total image size even for a relatively fast test sequence like *Foreman*. Even for *Stefan* sequence, which has a very high motion, the interpolation errors are less than 8% of the total image size. This enables us to apply a suitable interpolation technique to conceal these errors.

Consider a sub-image of an original frame of the sequence *Foreman* and its 3-D surface plot, as shown in Fig. 5.1 (a). In the compensated frame, we locate the positions of the overlapped pixels and the unfilled holes, and assign a zero value to these pixels. Fig. 5.1(b) shows the sub-image in the compensated frame with the unknown pixels, and its 3D surface plot. We shall refer, hereafter, to such overlapped pixels and unfilled holes as the unknown pixels. The remaining pixels are referred to as known pixels. From the 3D surface plot of the sub-image shown in Fig. 5.1(b), it is seen that the surface of the compensated frame is not continuous due to the presence of the unknown pixels. Our goal is to find a suitable function that can assign a value to each of the unknown pixels, such that the 3D surface plot of the sub-image of the interpolated frame has a continuous surface similar to that shown in Fig. 5.1(a). Once such a value is assigned to an unknown pixel, the resulting pixel will be referred to as a restored pixel and the corresponding assigned pixel value as the restored pixel value. Since in typical video images, the luminance value does not change abruptly, we can conceal the interpolation errors so that the following requirements are satisfied: (a) the restored pixels are smoothly connected

with the adjacent pixels, (b) the dominant edges are preserved as much as possible, and (c) the values of the known pixels are retained.

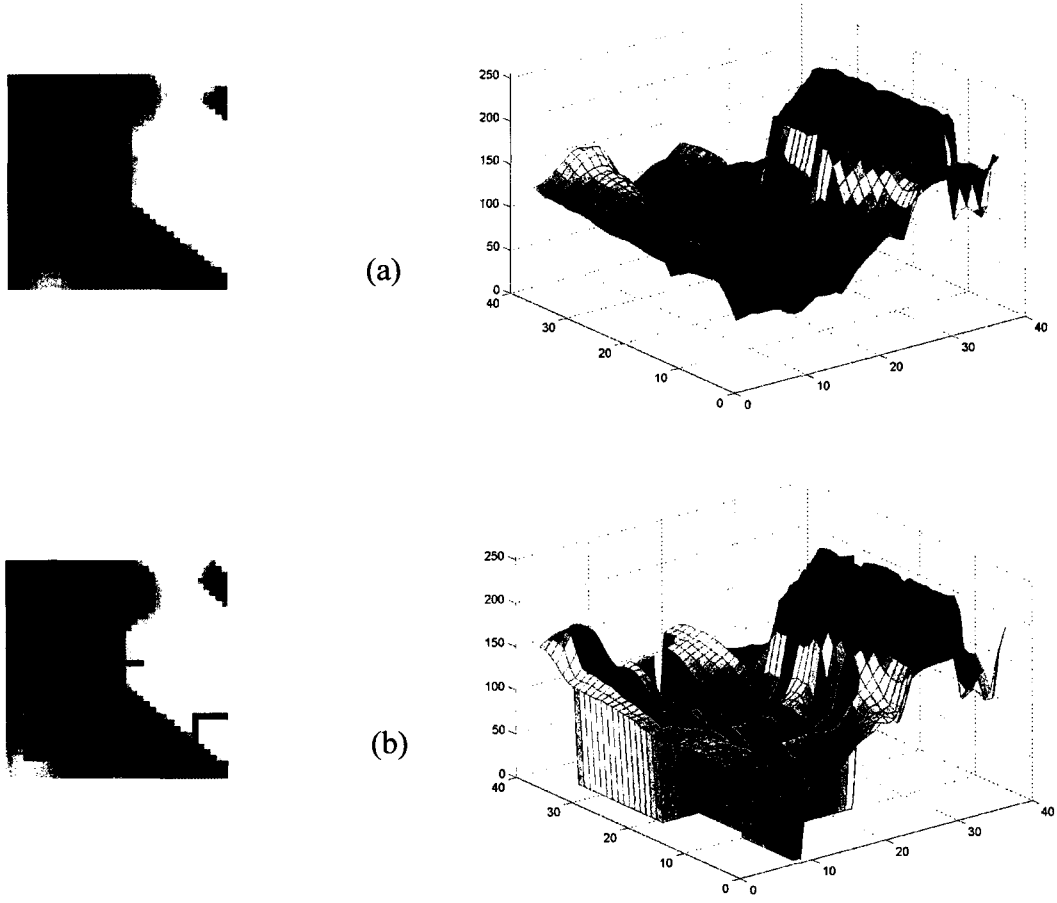


Figure 5.1: (a) A sub-image of an original frame of the sequence *Foreman* and its 3D surface plot. (b) The corresponding sub-image in the compensated frame, with the interpolation errors, and its 3D surface plot

The proposed interpolation error concealment MCI scheme is shown in Fig. 5.2. It consists of a module selector, a generic iterative refinement (GIR) module and an edge-reuniting iterative refinement (ERIR) module. The module selector is utilized to determine as to which specific interpolation module, GIR or ERIR, is to be used for the concealment of the interpolation errors. If there is no dominant edge detected, then the GIR module, which is capable of presenting a rather smooth reconstruction in the

homogeneous area, is employed. If there is a dominant sharp edge detected in the image, then the ERIR module is utilized for the error concealment in the direction of the detected edge.

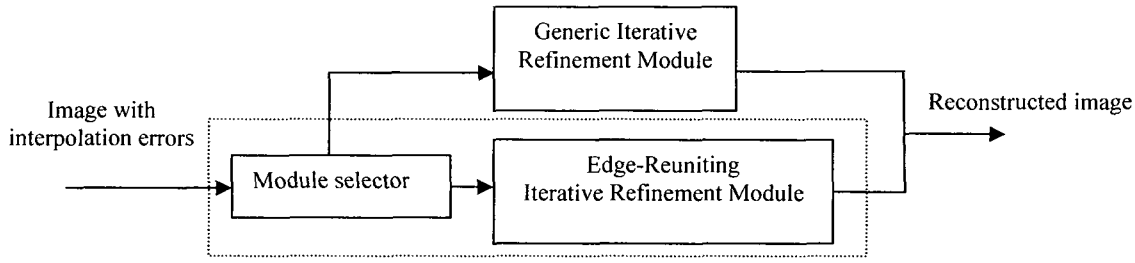


Figure 5.2: Proposed interpolation error concealment scheme for MCI.

5.2.1 Module selector

The main task of the module selector is to choose the appropriate interpolation method according to the result of the gradient estimator. We apply a gradient estimator to the $F\hat{R}M_{t-\alpha}$. Thus, for an image block BLK_k $(BLK_k \in F\hat{R}M_{t-\alpha})$ that has unknown pixels inside, we can determine if there is a dominant sharp edge with a certain orientation. Either GIR or ERIR is selected based on the results. If BLK_k does possess a dominant sharp edge, the ERIR is employed to conceal the interpolation errors; otherwise, GIR is utilized for the processing.

In Section 4.3, we used the Roberts operator as the gradient estimator and presented a voting mechanism in order to determine if there exists a dominant edge in an image block. We now apply the same techniques to the module selector, so as to detect the dominant sharp edge in BLK_k. With the help of the Roberts operator, the angular direction and magnitude of the gradient of a known pixel at (i, j) in the block BLK_k of the frame $F\hat{R}M_{t-\alpha}$ are determined. For each known pixel at (i, j), we first obtain

$$g_x = \hat{F}_{t-\alpha}(i,j) - \hat{F}_{t-\alpha}(i+1,j+1) \tag{5.1}$$

$$g_y = \hat{F}_{t-\alpha}(i,j+1) - \hat{F}_{t-\alpha}(i+1,j) \tag{5.2}$$

Then, the magnitude $G(i,j)$ and the angular direction $\theta(i,j)$ of the gradient at (i,j) can be calculated by using (4.19) and (4.20). After carrying out the directional classification of known pixels, and performing the voting process for each angular direction, we obtain the ψ_D, which is associated with R_{Max}. This register R_{Max} has the largest value as the result of the counting process, which increments a certain register R_r by $G(i,j)$ for each of the known pixels. Most of the time, ψ_D represents the orientation for the dominant edge. However, in the case when there are several edge orientations detected in BLK_k, and the register associated with an orientation has a value quite comparable to another register, it just reflects the fact that BLK_k contains more than one discernible edge. Thus, if the interpolation is performed along only a given orientation, this may lead to undesirable results. The module selector determines as to which module to switch to, based on the following criterion, similar to that defined in Section 4.3.

If for any $R_r \neq R_{\max}$, and $R_r < 0.2 \cdot R_{\max}$, then ψ_D is considered as the orientation for a dominant edge, and ERIR is carried out along ψ_D. Otherwise, the module selector determines that BLK_k has no dominant edge and the GIR module is employed to conceal the interpolation errors.

5.2.2 Generic iterative refinement

The generic iterative refinement (GIR) module is based on the image interpolation model derived in Chapter 3. This module aims at concealing the interpolation errors in

homogeneous areas, so that the interpolated pixels are smoothly connected with the adjacent pixels. In order to interpolate the unknown pixels in $F\hat{R}M_{t-\alpha}$, we construct a continuous function $F^c_{t-\alpha}(x,y)$, the domain Λ of this function being a given frame. Therefore, for frame of size $(W \times H)$ pixels, Λ is

$$\Lambda = [0, W-1] \times [0, H-1] \tag{5.3}$$

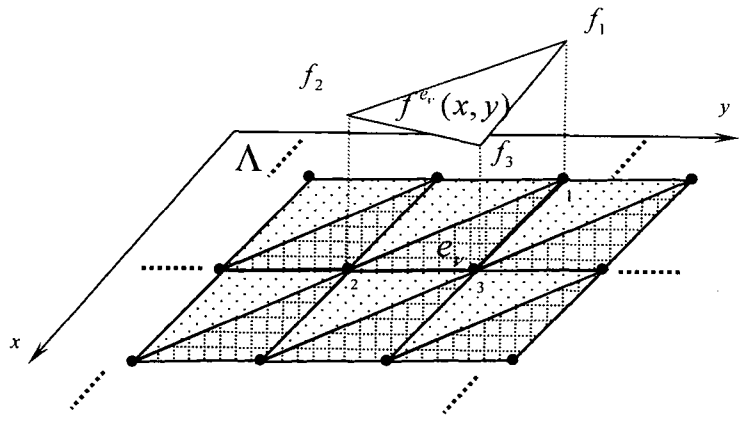
where W and H denote the width and height of the video frame. As presented in Section 3.2.1, we employ triangular segmentation and partition Λ into η $(\eta \in Z)$ non-overlapping triangular elements e_ν $(\nu = 1, \cdots, \eta)$, the nodes of which correspond to the pixel locations in the frame, as shown in Fig. 5.3(a). These elements are then classified into two types of local triangular elements, the first, second, and third nodes of which are ordered counterclockwise (see Figs. 5.3(b) and 5.3(c)). Γ is the set containing all the nodes in Λ. That is, $\Gamma \subset \Lambda$,

$$\Gamma = (0,1,\ldots,W-1) \times (0,1,\ldots,H-1) \tag{5.4}$$

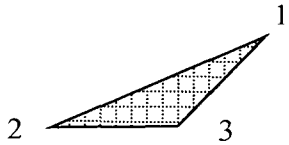
After the triangular segmentation of the frame $F\hat{R}M_{t-\alpha}$, the nodes corresponding to the known pixels are referred to as the known nodes, whereas the remaining nodes as unknown nodes. Let K $(\mathrm{K} \subset \Gamma)$ denote the set of known nodes and U $(U \subset \Gamma)$ the set of unknown nodes, $U \bigcup \mathrm{K} = \Gamma$ and $U \bigcap \mathrm{K} = \varnothing$. Under the condition

$$F^c_{t-\alpha}(x,y) = \hat{F}_{t-\alpha}(x,y), \ (x,y) \in \mathrm{K} \tag{5.5}$$

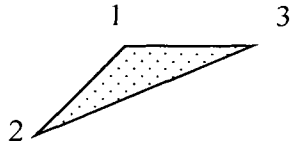
let us express $F^c_{t-\alpha}(x,y)$ through the use of the triangular element, so that the function is characterized by the element's nodes



(a)



(b)



(c)

Figure 5.3: (a) The triangular elements partitioned in the domain Λ of $F^c_{t-\alpha}(x,y)$. Shaded area is the restored luminance function in the element e_ν. (b) Type I local triangular element. (c) Type II local triangular element.

$$F^c_{t-\alpha}(x,y) = f^{e_\nu}(x,y) \quad (x,y) \in e_\nu \tag{5.6}$$

where

$$f^{e_\nu}(x,y) = \begin{pmatrix} f_1 & f_2 & f_3 \end{pmatrix} \begin{pmatrix} \vartheta_1(x,y) \\ \vartheta_2(x,y) \\ \vartheta_3(x,y) \end{pmatrix} \tag{5.7}$$

$f^{e_\nu}(x,y)$ is a continuous function in e_ν, as shown in the shaded area of Fig. 5.3(a), with

$$f_i = F^c_{t-\alpha}(x_i,y_i), \quad (x_i,y_i) \in \Gamma, \ (i = 1,2,3) \tag{5.8}$$

(x_i,y_i) being the location of the ith node in e_ν. The shape functions are given by

$$\vartheta_1(x,y)=\frac{1}{2s}\begin{vmatrix}1 & x & y\\ 1 & x_2 & y_2\\ 1 & x_3 & y_3\end{vmatrix} \tag{5.9}$$

$$\vartheta_2(x,y)=\frac{1}{2s}\begin{vmatrix}1 & x_1 & y_1\\ 1 & x & y\\ 1 & x_3 & y_3\end{vmatrix} \tag{5.10}$$

$$\vartheta_3(x,y)=\frac{1}{2s}\begin{vmatrix}1 & x_1 & y_1\\ 1 & x_2 & y_2\\ 1 & x & y\end{vmatrix} \tag{5.11}$$

with s denoting the area of the element e_ν. To restore the unknown nodes in $F\hat{R}M_{t-\alpha}$, $F^c_{t-\alpha}(x,y)$ should satisfy the requirements set earlier regarding the smooth fit of a restored pixel; hence, we apply the optimization constraint that the surface represented by the constructed $F^c_{t-\alpha}(x,y)$ has the smallest area. Then, we can find suitable values for the unknown nodes so as to minimize the function

$$\iint_{\Lambda}\sqrt{1+\left(\frac{\partial F^c_{t-\alpha}(x,y)}{\partial x}\right)^2+\left(\frac{\partial F^c_{t-\alpha}(x,y)}{\partial y}\right)^2}\,dxdy \tag{5.12}$$

By substituting (5.6) into (5.12), the above process is equivalent to finding a set containing $F^c_{t-\alpha}(i,j)$ at each node $(i,j)\in U$, which minimizes the function

$$\sum_{\nu=1}^{\eta}\iint_{E_\nu}\left[\left(\frac{\partial f^{E_\nu}(x,y)}{\partial x}\right)^2+\left(\frac{\partial f^{E_\nu}(x,y)}{\partial y}\right)^2\right]dxdy \tag{5.13}$$

It is noted that (5.13) has the same form as (3.6), and by using the FEM-based image interpolation model developed in Section 3.2, it can be shown that the optimal value of an unknown node at $(i,j)\in U$ satisfies the following:

$$F^c_{t-\alpha}(i,j) = \beta_V(i,j) + \beta_H(i,j) \tag{5.14}$$

with

$$\beta_V(i,j) = \begin{cases} \frac{1}{2}F^c_{t-\alpha}(i+1,j) & if\ i = 0 \\ \frac{1}{2}F^c_{t-\alpha}(i-1,j) & if\ i = H-1 \\ \frac{1}{4}\left[F^c_{t-\alpha}(i-1,j) + F^c_{t-\alpha}(i+1,j)\right] & if\ i \neq 0 or H-1 \end{cases}$$

and

$$\beta_H(i,j) = \begin{cases} \frac{1}{2}F^c_{t-\alpha}(i,j+1) & if\ j = 0 \\ \frac{1}{2}F^c_{t-\alpha}(i,j-1) & if\ j = W-1 \\ \frac{1}{4}\left[F^c_{t-\alpha}(i,j+1) + F^c_{t-\alpha}(i,j-1)\right] & if\ j \neq 0 or W-1 \end{cases}$$

It can be seen that (5.14) has the same form as (3.34). Therefore, in order to obtain the optimal value for an unknown node $(i,j) \in U$, a set of linear equations needs to be solved. In view of the number of equations being very large, we employ the iterative refinement procedure, by using the following equations, which are similar to (3.35).

$$F^{c\ (k+1)}_{t-\alpha}(i,j) = \beta^{(k)}_V(i,j) + \beta^{(k)}_H(i,j), \qquad (k \geq 0) \tag{5.15}$$

with

$$\beta^{(k)}_V(i,j) = \begin{cases} \frac{1}{2}F^{c\ (k)}_{t-\alpha}(i+1,j) & if\ i = 0 \\ \frac{1}{2}F^{c\ (k+1)}_{t-\alpha}(i-1,j) & if\ i = H-1 \\ \frac{1}{4}\left[F^{c\ (k+1)}_{t-\alpha}(i-1,j) + F^{c\ (k)}_{t-\alpha}(i+1,j)\right] & if\ i \neq 0 or H-1 \end{cases}$$

and

$$\beta_H^{(k)}(i,j)=\begin{cases}\dfrac{1}{2}F_{t-\alpha}^{c\ (k)}(i,j+1) & if\ j=0\\ \dfrac{1}{2}F_{t-\alpha}^{c\ (k+1)}(i,j-1) & if\ j=W-1\\ \dfrac{1}{4}\left[F_{t-\alpha}^{c\ (k+1)}(i,j-1)+F_{t-\alpha}^{c\ (k)}(i,j+1)\right] & if\ j\neq 0 or W-1\end{cases}$$

This iterative refinement procedure is carried out in a raster scan order, and applied to the unknown nodes in Λ. It is noted that if $(x,y)\in \mathrm{K}$, then $F_{t-\alpha}^{c\ (k+1)}(x,y)=F_{t-\alpha}^{c\ (k)}(x,y)$, and hence, the values of the known nodes involved in (5.15) remain unchanged. When the given stop criterion is met, we obtain the luminance value of a restored pixel at $(i,j)\in U$. When all the unknown nodes in Λ are assigned such values, we have the interpolated frame $FRM_{t-\alpha}$. In the regions, where interpolation errors have occurred, the restored pixels will exhibit a smooth variation without any discontinuity.

Let us now formulate the procedure for the GIR, which is applied for the concealment of interpolation errors in $F\hat{R}M_{t-\alpha}$. Let $\vec{f}$ be a vector, each component of which corresponds to the computed value of an unknown pixel $(i,j)\in U$, and $F_{t-\alpha}^{c\ (0)}(i,j)$ be the initial value of such a pixel. Let ε be a pre-specified tolerance. Then, the GIR is carried out as follows.

(1) Initialize $\vec{f}^{(0)}$ with $F_{t-\alpha}^{c\ (0)}(i,j)$, and set $k=0$.

(2) For each pixel at $(i,j)\in U$, compute $F_{t-\alpha}^{c\ (k+1)}(i,j)$ using (5.15).

(3) If $MAPD(\vec{f}^{(k+1)},\vec{f}^{(k)})<\varepsilon$, set $\hat{\vec{f}}=\vec{f}^{(k+1)}$, and stop.

(4) Otherwise, increment k and go to step 2.

In the above, $\hat{\vec{f}}$ represents the vector of restored pixel values. The operation of $MAPD(\cdot,\cdot)$ is restricted to integer accuracy so that we can set the tolerance $\varepsilon = 0$.

Since the algorithm above can be applied to an image of size $M \times N$ $(M > 1, \quad N > 1)$, we can partition the compensated frame into non-overlapping blocks, BLK_i, $i = 1,2,\ldots,Q$, $Q = (W \times H)/(M \times N)$, so that the computational effort to conceal the interpolation errors gets distributed among the various blocks having interpolation errors.

5.2.3 Edge-reuniting iterative refinement

When the module selector determines that ψ_D is the orientation for a dominant edge, the ERIR should be carried out along ψ_D. Based on the fact that along an edge direction, the pixel values change more slowly than they do across an edge, the pixel continuity should be maintained along the edge by ERIR. Under this condition, the GIR can be adapted to the ERIR. Given the fact that ψ_D is the orientation of the detected sharp dominant edge in BLK_k, an unknown pixel at (i, j) should be restored along the detected edge ψ_D. Let us consider the following different cases.

If $\psi_D = 0°$, the interpolation for an unknown pixel in BLK_k is carried out in the vertical direction. Based on (5.14), we have

$$F^c_{t-\alpha}(i,j) = \beta_V(i,j) \tag{5.16}$$

If $\psi_D = 90°$, the interpolation is carried out in the horizontal direction, and

$$F^c_{t-\alpha}(i,j) = \beta_H(i,j) \tag{5.17}$$

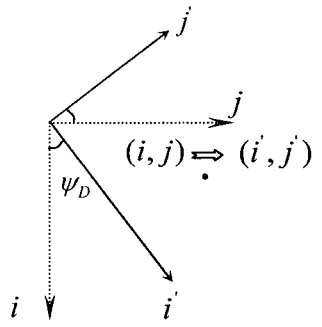


Figure 5.4: An unknown pixel in the two coordinate systems

When ψ_D is neither 0° nor 90°, in order to find the immediate neighbor of an unknown pixel along ψ_D or in a direction perpendicular to ψ_D, we rotate the coordinate system in counter-clockwise direction by ψ_D. Let (i, j) and (i', j') represent the coordinate of an unknown pixel in the original and new coordinate systems, respectively. This is shown in Fig. 5.4, where the dotted lines represent the original coordinate system and the solid lines the new coordinate system.

The immediate neighbor of an unknown pixel in the new coordinate system is given by $(i'+l, j'+q)$, $(l = -1, 0, 1; q = -1, 0, 1; l \neq q)$. Its counterpart in the original coordinate system can be obtained from the following equation, which has the same form as (4.25)

$$\begin{pmatrix} i'+l \\ j'+q \end{pmatrix} = \begin{pmatrix} i \\ j \end{pmatrix} + \begin{pmatrix} l & -q \\ q & l \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} \tag{5.18}$$

where

$$\Delta x = \arg\min_h \{ [h \cdot \tan(\psi_D)]_{round} > 0 \} \quad h \text{ an integer} \tag{5.19}$$

$$\Delta y = [\Delta x \cdot \tan(\psi_D)]_{round} \tag{5.20}$$

Hence, the optimal value for an unknown pixel A at (i, j) is derived from (5.14) as

$$F_{t-\alpha}^{c}(i,j) = 2\beta_V(i,j) \quad \psi_D \neq 0^\circ,\ 90^\circ \tag{5.21}$$

The ERIR is performed in the orientation ψ_D of a detected dominant edge. Equations (5.16), (5.17) and (5.21) can be utilized, in an iterative manner, to interpolate an unknown node $(i, j) \in U$. This ERIR module is characterized by its capability to maintain a smooth variation along the detected direction ψ_D. As will be shown in Section 5.3, this module can properly reunite a broken edge, which is caused by unknown nodes.

Hereafter, we refer to the MCI scheme, which utilizes all the 3 modules, as the ERIR-MCI scheme. In a situation, where the image is quite smooth or the computational complexity is of a greater concern than the reconstructed image quality, the module selector and ERIR module can be skipped. Then, the resulting scheme is referred to as the GIR-MCI scheme.

5.3 Simulation Results

In order to evaluate the schemes that have been developed in the previous section, several original 30 frames/s test sequences are sub-sampled to 15 frames/s (every other frame in the sequence is skipped) and 10 frames/s (2 out of 3 frames are skipped), respectively. Two experiments are carried out to test the performance of the proposed schemes. In the first experiment, all the skipped frames are interpolated using the sub-sampled test sequences and the motion vector field generated through the full search block matching algorithm applied to the sequences at 15 frames/s and 10 frames/s respectively. This is to study the performance of the proposed interpolation scheme by excluding the effect of

the artifacts in the decoded frames. The second experiment involves simulations carried out with H.264/AVC reference software JM15.0 [69]. The test sequences are encoded at 15frames/s and 10 frames/s, respectively, and the proposed schemes implemented in the decoder. All the skipped frames are interpolated using the decoded frames and motion vector fields available at the decoder. This is to study as to how our scheme performs, when used in a video decoder. In this experiment, the parameters, quantization parameter (QP)=24, and P-frame/I-frame (P/I) ratio=10, of the JM15.0 encoder remain unchanged, so that the decoded frames have the same quality across all the different schemes.

In each experiment, the six schemes, FR, SMVF [43], MCI+FR which predicts the unfilled pixels with FR [44], and DB-FMCI [38], along with the proposed ERIR-MCI scheme, and the proposed GIR-MCI scheme that conceals the interpolation errors by only implementing the generic iterative refinement module. It is to be noted that DB-FMCI includes pixel classification, moving objects segmentation, and affine transform, and hence, should be considered as a sophisticated MCI scheme with a high computational complexity. In spite of this, we implement the sophisticated DB-FMCI in the experiments, in order to evaluate the proposed schemes with respect to this scheme in terms of both the PSNR and visual performance. The experiments are carried out on a Pentium Celeron 2.4 GHz PC, using eight standard video sequences: frames 90 to 180 of *Foreman* (QCIF), frames 20 to 110 of *Suzie* (QCIF), frames 20 to 110 of *Miss America* (QCIF), frames 40 to 130 of *Mother & Daughter* (QCIF), frames 1 to 90 of *Stefan* (CIF), frames 20 to 110 of *Coast Guard* (CIF), frames 10 to 100 of *Container* (CIF), and frames 30 to 120 of *News* (CIF).

In the ERIR-MCI or GIR-MCI method, the predicted frame is partitioned into blocks for the purpose of distributing the computational load among several blocks. To examine the performance resulting from different block sizes, we divide a QCIF frame (176×144 pixels) into non-overlapping image blocks of sizes 8×8 pixels, 16×16 pixels, and 48×44 pixels, respectively. Under the same conditions (initialized with zero, interpolated with original frames), simulation results reveal that a larger block size leads to better performance in terms of both the subjective and objective qualities; however, the number of basic operations (including those of addition, multiplication, and absolute difference) increases. Results of the GIR-MCI method implemented with the *Foreman* test sequence are shown in Table 5.3. A block size of 16×16 is chosen for the proposed schemes, since it coincides with the macro-block size employed in most conventional coding schemes of the video coding standards. At the same time, better objective and subjective performances are achieved when the block size used is 16×16 rather than a block size of 8×8. The initialization value $F_{t-\alpha}^{c}{}^{(0)}(i,j)$ has an influence on the speed of convergence. We choose for $F_{t-\alpha}^{c}{}^{(0)}(i,j)$ three different values, namely, 0, the minimum luminance value and the mean luminance value of the known pixels in the image block. Again, the GIR-MCI method is implemented with the *Foreman* test sequence to select the best initialization value, and the results are given in Table 5.4. The table shows that an initialization with the mean value of the known pixels results in the fastest convergence, leading to the smallest number of basic operations. Simulation results show that with such an initialization value, the average number of iterations needed to reach a stable value depends on the sequence under consideration. For example, for fast sequences such as the *Foreman* test sequence, about 6 to 7 iterations on average are needed, whereas for

sequences without fast motion, such as *Miss America* or *Mother & Daughter*, the average number of iterations is about three.

Table 5.3: Computational complexity and PSNR resulting from the use of various block sizes with QCIF *Foreman* sequence (using GIR-MCI scheme)

Block size ($M \times N$ pixels)	8× 8	16× 16	48× 44
Basic operations	55339	64205	66935
PSNR (dB)	29.86	29.98	30.05

Table 5.4: Comparison of computational complexity with different initialization values with QCIF *Foreman* sequence, image block size 16×16 (using GIR-MCI scheme)

Initialization value	0	Minimum value of known pixels	Mean value of known pixels
No. of basic operations	64205	49878	28407

Table 5.5 and Table 5.6 present the performance results for the various schemes in terms of the average PSNR values, for interpolation with the original image sequences and JM15.0 H.264 decoded frames at 15 frames/s, whereas Table 5.7 and Table 5.8 present the corresponding results when implemented with sequences at 10 frames/s. The average PSNR is obtained over all the interpolated frames. It is observed that MCI+FR, ERIR-MCI and GIR-MCI provide a performance that is better than that of the simple temporal interpolation technique such as the FR technique. The ERIR-MCI and GIR-MCI schemes exhibit a performance comparable to that of MCI + FR scheme, and all the three schemes are superior to the SMVF scheme. Not

surprisingly, the sophisticated DB-FMCI scheme presents the best PSNR results among all the 6 schemes; however, this is achieved at a very high computational cost, and this point will be addressed later in this section.

Table 5.5: Average PSNR values (dB) using various schemes (tested with 15 frames/s image sequences, original frames)

Original frames (15 frames/s)	FR	SMVF	MCI+FR	GIR-MCI	ERIR-MCI	DB-FMCI
Mother&Daughter	39.11	39.01	39.12	39.11	39.13	40.07
Miss America	38.48	37.93	39.54	39.53	39.52	40.63
Suzie	30.18	30.53	32.42	32.56	32.6	33.61
Foreman	28.62	28.63	30.02	29.98	30.09	30.93
Container	37.36	37.54	38.32	38.3	38.31	39.23
News	34.84	34.95	35.02	35.08	35.1	35.94
Coast Guard	23.78	23.87	27.28	27.3	27.31	27.97
Stefan	19.96	20.8	22.8	22.95	22.97	23.52

Table 5.6: Average PSNR values (dB) using various schemes (tested with 15 frames/s image sequences, decoded frames)

Decoded frames (15 frames/s)	FR	SMVF	MCI+FR	GIR-MCI	ERIR-MCI	DB-FMCI
Mother&Daughter	38.53	38.42	38.54	38.52	38.56	39.29
Miss America	37.77	37.21	38.84	38.85	38.83	39.57
Suzie	29.51	29.79	31.76	31.88	31.95	32.53
Foreman	27.96	26.43	28.96	28.92	29.05	29.57
Container	34.32	34.93	35.37	35.33	35.35	35.95
News	33.7	33.89	34.18	34.23	34.25	34.8
Coast Guard	23.65	23.79	26.72	26.71	26.73	27.21
Stefan	19.82	20.65	22.55	22.63	22.66	23.09

Table 5.7: Average PSNR values (dB) using various schemes (tested with 10 frames/s image sequences, original frames)

Original frames (10 frames/s)	FR	SMVF	MCI+FR	GIR-MCI	ERIR-MCI	DB-FMCI
Mother&Daughter	37.15	37.45	37.56	37.74	37.76	38.63
Miss America	37.26	37.17	39.14	39.13	39.12	40.22
Suzie	28.97	29.61	31.12	31.26	31.3	32.24
Foreman	27.19	27.77	29.72	29.68	29.79	30.59
Container	33.37	34.46	37.9	37.87	37.88	38.79
News	30.14	32.99	34.23	34.35	34.39	35.18
Coast Guard	20.87	22.59	26.59	26.96	26.98	27.6
Stefan	19.08	20.69	21.21	21.6	21.61	22.11

Table 5.8: Average PSNR values (dB) using various schemes (tested with 10 frames/s image sequences, decoded frames)

Decoded frames (10 frames/s)	FR	SMVF	MCI+FR	GIR-MCI	ERIR-MCI	DB-FMCI
Mother&Daughter	36.6	36.88	36.99	36.97	37.01	37.68
Miss America	36.26	36.1	38.07	38.08	38.06	38.75
Suzie	28.33	28.6	30.49	30.61	30.68	31.2
Foreman	26.57	25.37	28.39	28.35	28.48	28.96
Container	32.15	33.74	35.31	35.28	35.3	35.86
News	29.92	32.44	33.47	33.56	33.6	34.14
Coast Guard	20.79	22.52	26.05	26.57	26.4	26.88
Stefan	19.07	20.54	20.85	21.37	21.38	21.76

In order to compare the performance of the proposed schemes from the point of view of the visual quality, we consider frames 178, 179 and 180 of the video sequence *Foreman*, in which there is a significant motion in the head and shoulder areas of the sequence. To highlight the simulation results, we present the results for the face portion of the frame 179 interpolated with the original frames and JM15.0 H.264 decoded frames in Fig. 5.5 and Fig. 5.6, respectively. As seen from Fig. 5.5, the FA scheme presents the worst results and this is due to its simple averaging strategy. It is observed from Fig. 5.5(c) and 5.5(d) that the SMVF scheme yields some blurred and block artifacts, whereas the MCI + FR scheme reveals unpleasant stripe-like artifacts in the interpolated frame. Since these annoying artifacts appear in the ROI, the subjective quality of the interpolated image severely deteriorates. In order to show how effective the proposed schemes are in concealing the interpolation errors, the face portion of the compensated frame with interpolation errors is shown in Fig. 5.5(e). The results are presented in Fig. 5.5(f) and Fig. 5.5(g), when GIR-MCI and ERIR-MCI schemes are, respectively, employed. It can be observed that both these schemes outperform MCI+FR, SMVF, and FA by providing a finer texture in the face portion. In Fig. 5.5 (h), the interpolated frame using sophisticated DB-FMCI is presented. It also provides good results in the face portion. However, by observing the mouth area in Fig. 5.5 (h), a distortion-like artifact can be seen. This is due to the fact that DB-FMCI utilizes the affine transform to reproduce the pixel-based motion vector from the block-based motion vector, and this could cause shape distortion to some degree. Similar observations can be observed from Fig. 5.6, in which, not surprisingly, some decoded artifacts can be noticed.

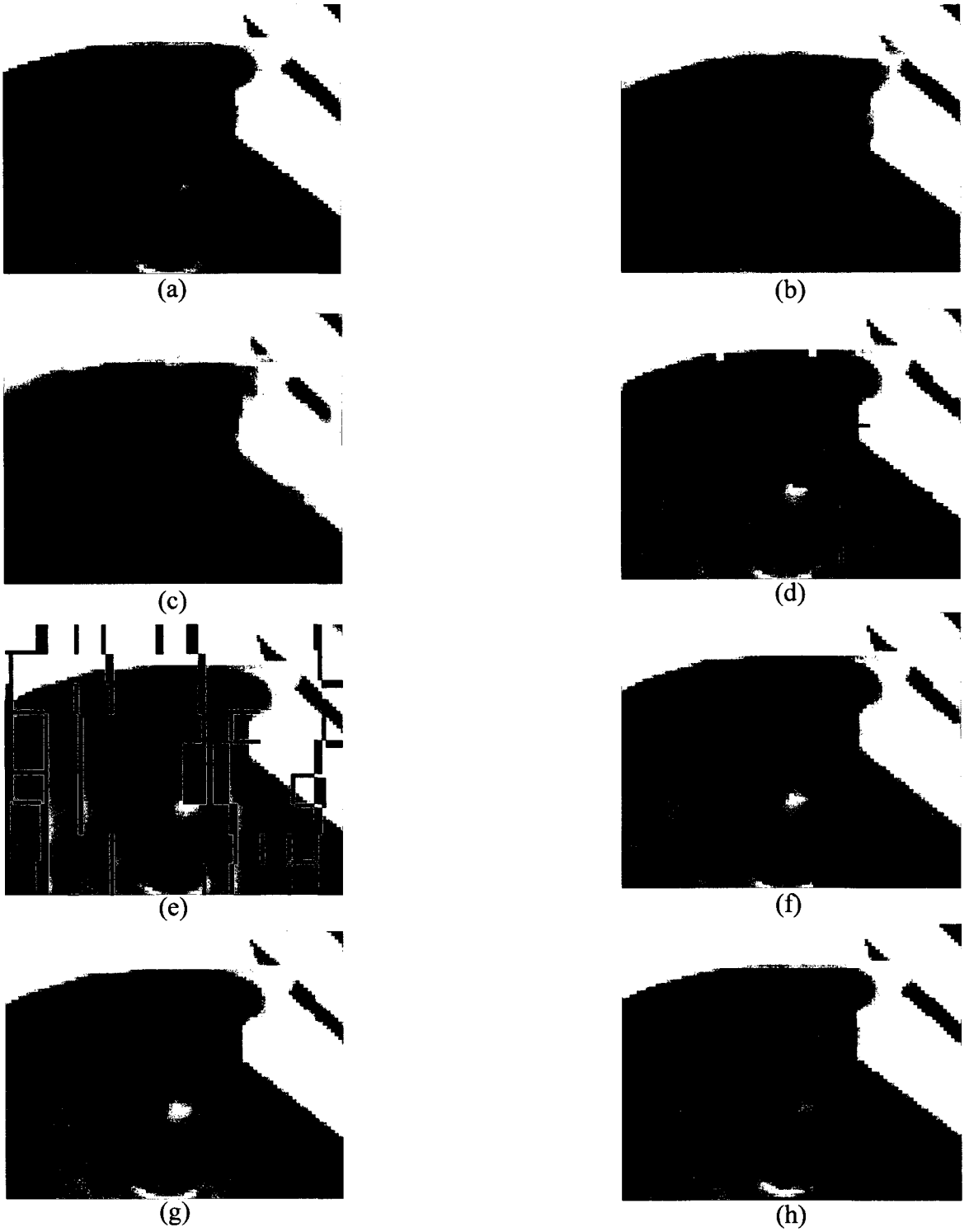


Figure 5.5: Original and interpolated frames (face portion only) using various schemes corresponding to frame 179 of the *Foreman* sequence. (Simulated with the original frames, 15 frames/s) (a) Original. (b) Interpolated frame using FA. (c) Interpolated frame using SMVF. (d) Interpolated frame using MCI+FR. (e) Predicted frame with interpolation errors. (f) Interpolated frame using GIR-MCI. (g) Interpolated frame using ERIR-MCI. (h) Interpolated frame using DB-FMCI

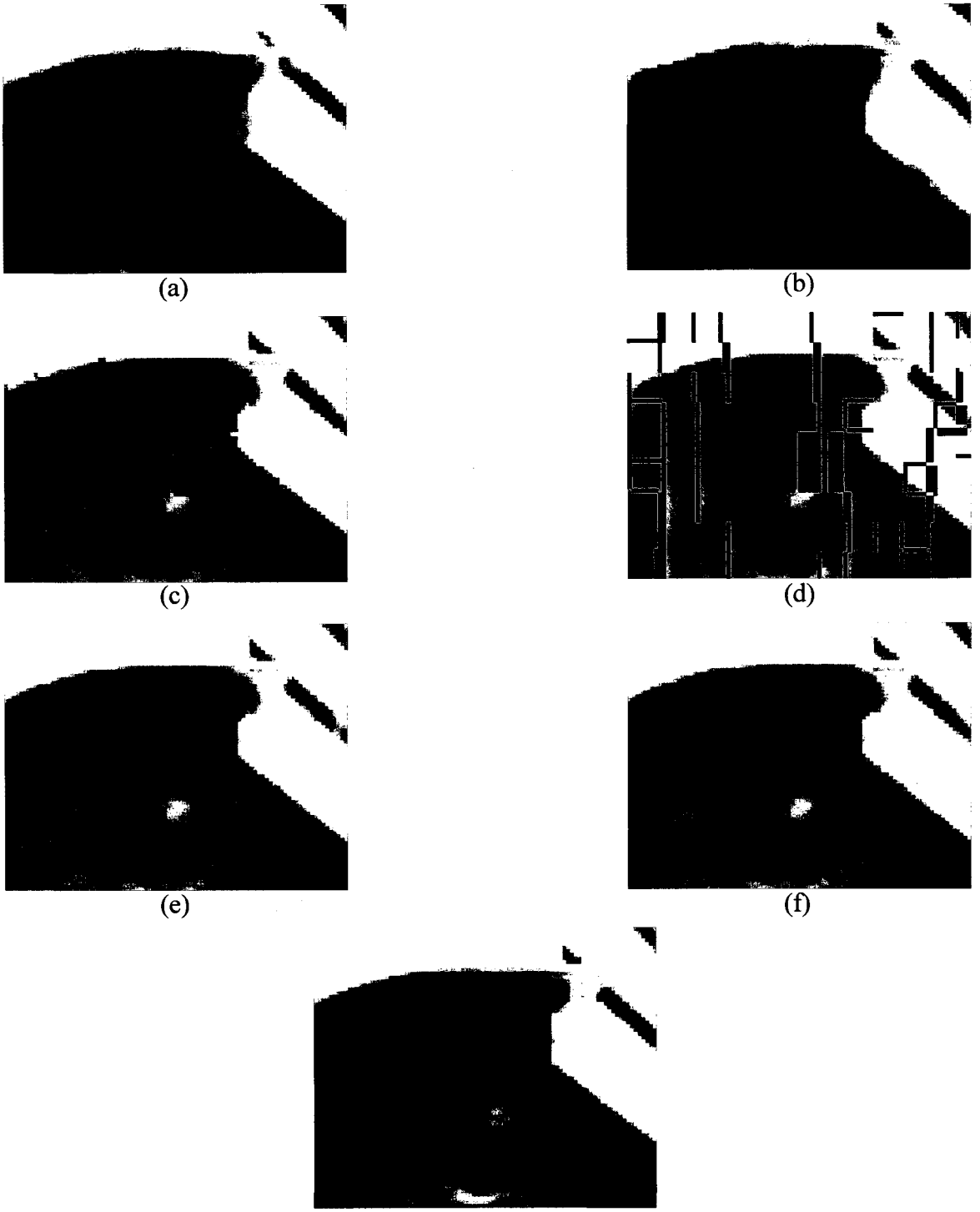


Figure 5.6: Interpolated frames (face portion only) using various schemes corresponding to frame 179 of the *Foreman* sequence. (Implemented with the JM15.0 H.264 decoded frames, 15 frames/s) (a) Interpolated frame using FA. (b) Interpolated frame using SMVF. (c) Interpolated frame using MCI+FR. (d) Predicted frame with interpolation errors. (e) Interpolated frame using GIR-MCI. (f) Interpolated frame using ERIR-MCI. (g) Interpolated frame using DB-FMCI.

A similar comparison is now made with an interpolated frame, namely, frame 22 from Stefan sequence, which has a very high motion. The results on the center portion of the interpolated frame are presented in Fig. 5.7 using the original frames. Similar to what was observed in Fig. 5.6, some blurred artifacts can be seen from the SMVF scheme in Fig. 5.7(c), and some annoying artifacts near the right shoulder area of the player can be observed with the MCI+FR scheme. Fig. 5.7(f) and Fig. 5.7(g) present the results for the GIR-MCI and ERIR-MCI schemes, respectively. The visual quality of the interpolated frame using the GIR-MCI or ERIR-MCI scheme is much better than that provided by MCI+FR and SMVF, and quite comparable to that of the DB-FMCI shown in Fig. 5.7 (h). By looking at the boundary line of the tennis court in the interpolated frame, it can be seen that the ERIR-MCI properly reunites the broken sharp edge caused by the interpolation errors, whereas the GIR-MCI shows some blurred effects on the edge. We discuss this advantage of the ERIR-MCI scheme over the GIR-MCI scheme in detail in the next paragraph. In Fig. 5.8 we present the results for interpolation with JM15.0 H.264 decoded frames. Again, similar conclusions can be drawn for the different schemes from this set of test results as well.

To better appreciate the advantage of the ERIR-MCI scheme in terms of maintaining the pixel continuity along the edges, we present in Fig. 5.9 and Fig. 5.10 the results on the background part of the interpolated frame 179 of Foreman. Fig. 5.9 shows the results obtained when the various schemes are implemented with the original frames, whereas Fig. 5.10 presents the results when implemented in the JM15.0 H.264 decoder. Both results are obtained with 15 frames/s image sequences. The interpolated frame using FA is shown in Fig. 5.9 (b), wherein the blurred effect in the edge area of the background

is observed. Some stripe effects may be seen in the edge area in the case of the SMVF scheme, as shown in Fig. 5.9(c). The MCI+FR scheme yields a relatively pleasant result as seen from Fig. 5.9(d), which is due to the fact that (i) it uses FR to deal with the unfilled pixels, and (ii) the similarity in the background part between two consecutive frames in the test sequence provides relatively accurate pixel values for the unfilled pixels. However, as mentioned above, it results in unpleasant stripe-like artifacts in the face portion. Fig. 5.9(e) presents the background part of the compensated frame with interpolation errors, and Fig. 5.9(f) the result obtained by the GIR-MCI scheme. We observe some blurred effects on the sharp edges with interpolation errors. In contrast, the ERIR-MCI scheme reunites the broken edges properly without causing any blurs, as seen from Fig. 5.9(g). This is achieved with the help of the module selector, which detects the dominant sharp edges in the interpolated frame, and switches to the ERIR mode, when necessary (see Fig. 5.2). From Fig. 5.10(f), it can be seen that ERIR-MCI works well when implemented in a H.264 decoder. It reunites the edges much better than the SMVF (Fig. 5.10(b)) or GIR-MCI (Fig. 5.10(e)) does.

The various schemes are also applied to the 10 frames/s original image sequences and H.264/AVC decoded image sequences respectively. Fig. 5.11 presents the simulation results on the face portion of the interpolated frame 179, using original image frames, whereas Fig. 5.12 presents the results obtained using H.264/AVC decoded frames. Fig. 5.13 presents the background part of the interpolated frame, using original frames, whereas the results obtained from the decoded frames are shown in Fig. 5.14. A visual comparison is also made with an interpolated frame, frame 21 from Stefan sequence through the use of 10 frames/s original image sequences and H.264/AVC decoded image

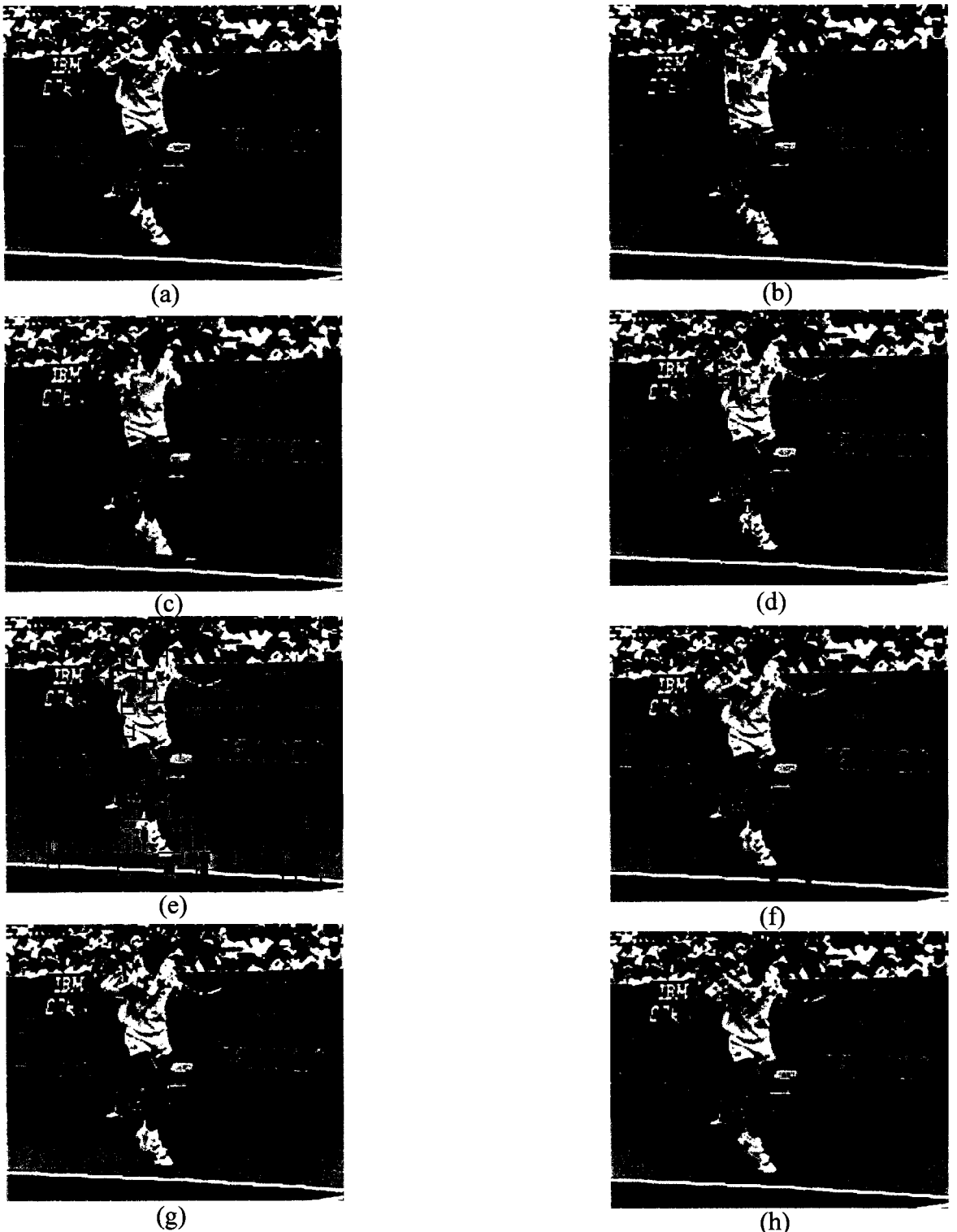


Figure 5.7: Original and interpolated frames (center portion) using various schemes corresponding to frame 22 of the CIF *Stefan* sequence. (Simulated with the original frames, 15 frames/s) (a) Original. (b) Interpolated frame using FA. (c) Interpolated frame using SMVF. (d) Interpolated frame using MCI+FR. (e) Predicted frame with interpolation errors. (f) Interpolated frame using GIR-MCI. (g) Interpolated frame using ERIR-MCI. (h) Interpolated frame using DB-FMCI.

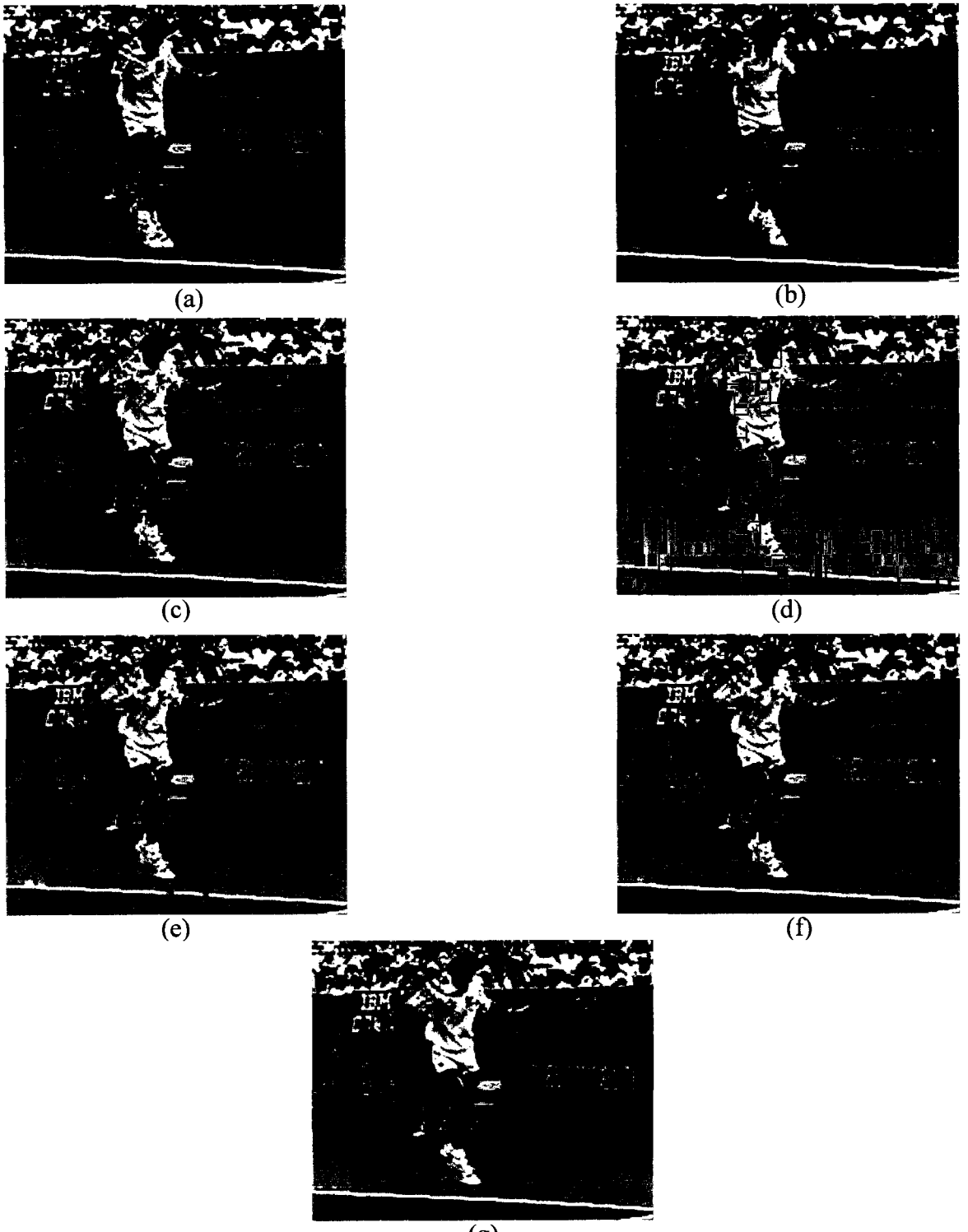


Figure 5.8: Interpolated frames (center portion) using various schemes corresponding to frame 22 of the CIF *Stefan* sequence. (Implemented with the JM15.0 H.264 decoded frames, 15 frames/s) (a) Interpolated frame using FA. (b) Interpolated frame using SMVF. (c) Interpolated frame using MCI+FR. (d) Predicted frame with interpolation errors. (e) Interpolated frame using GIR-MCI. (f) Interpolated frame using ERIR-MCI. (g) Interpolated frame using DB-FMCI.

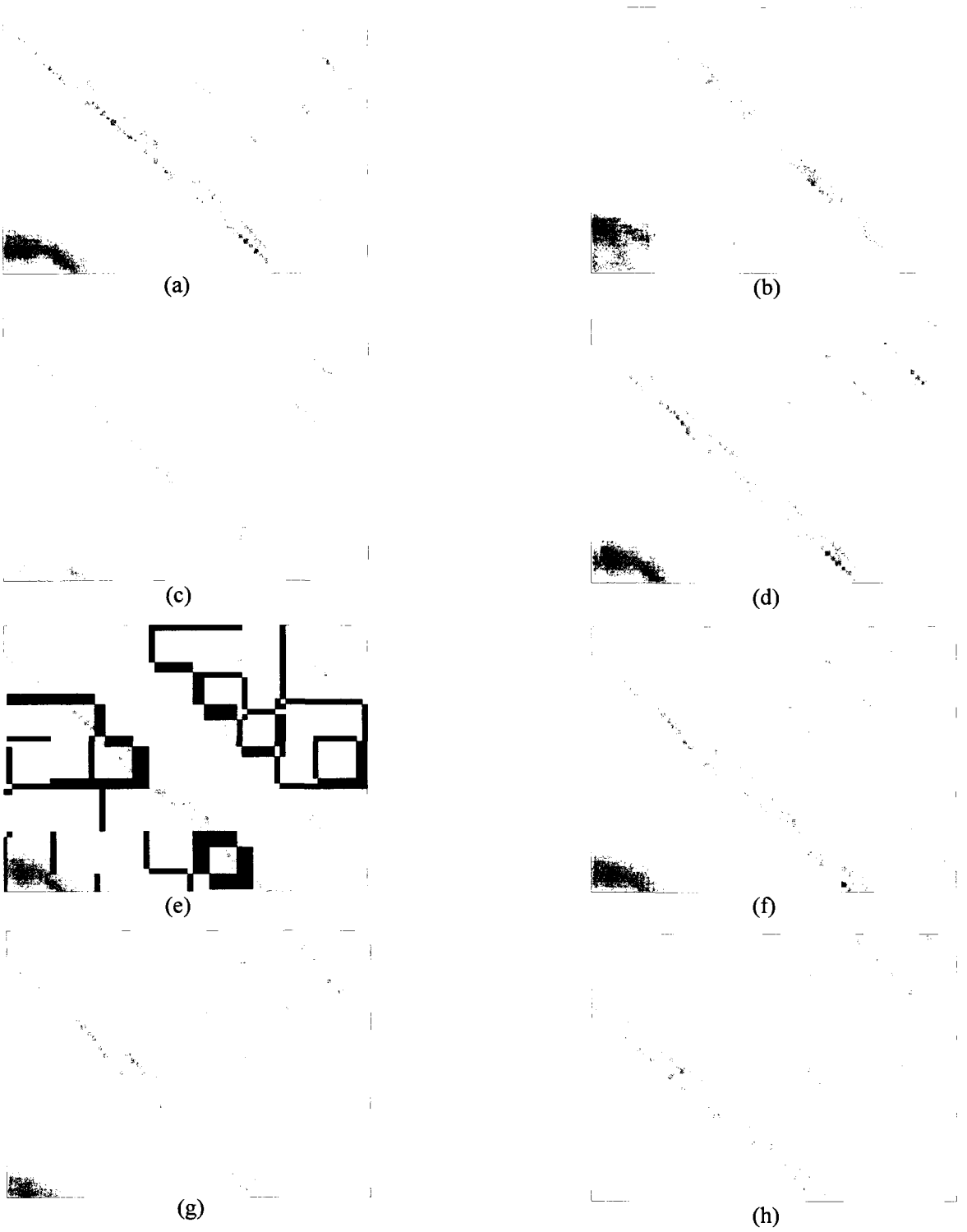


Figure 5.9: Original and interpolated frames (background part only) using various schemes corresponding to frame 179 of the *Foreman* sequence. (Simulated with the original frames, 15 frames/s) (a) Original frame. (b) Interpolated frame using FA. (c) Interpolated frame using SMVF. (d) Interpolated frame using MCI+FR. (e) Predicted frame with interpolation errors. (f) Interpolated frame using GIR-MCI. (g) Interpolated frame using ERIR-MCI. (h) Interpolated frame using DB-FMCI

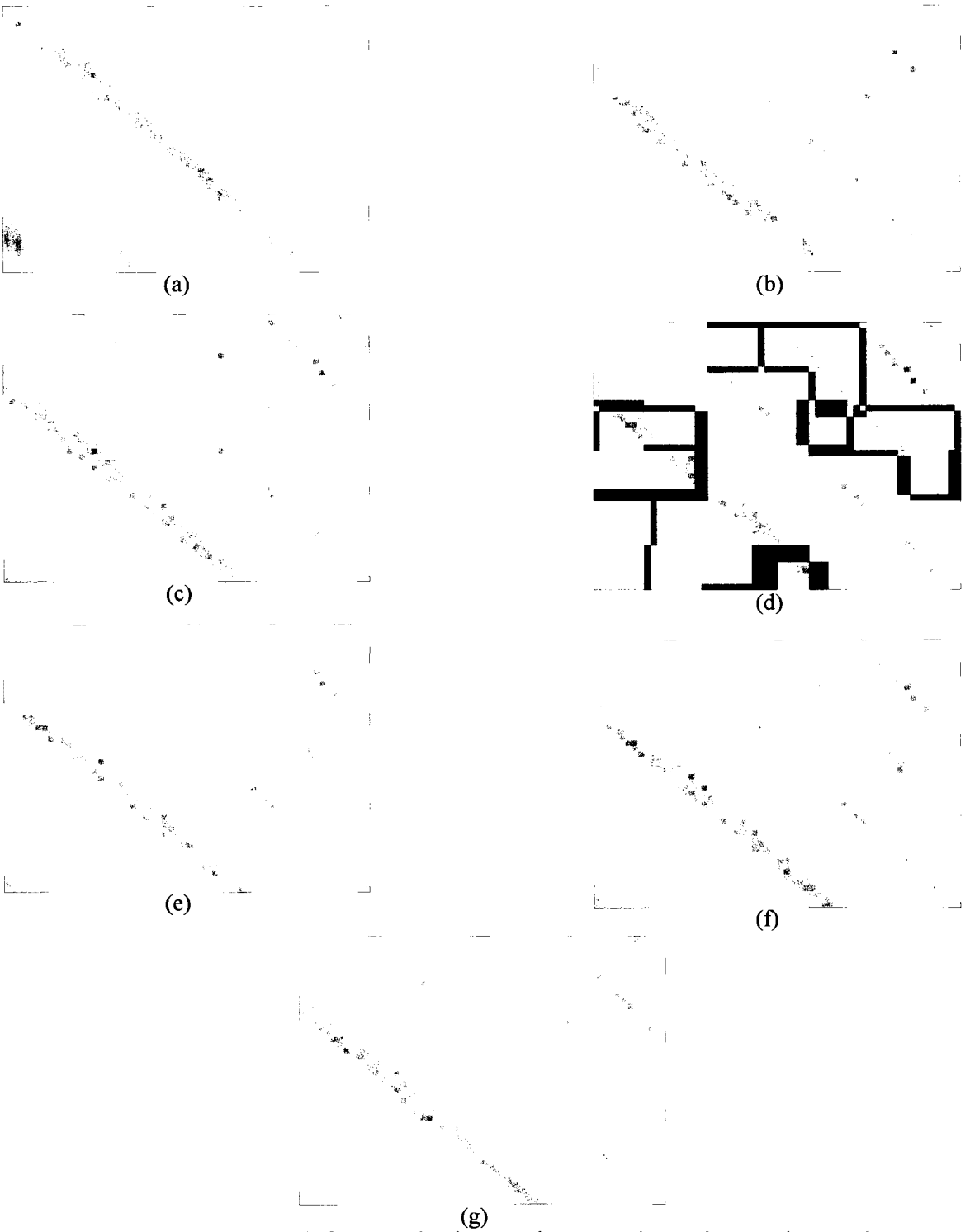


Figure 5.10: Interpolated frames (background part only) using various schemes corresponding to frame 179 of the *Foreman* sequence. (Implemented with the JM15.0 H.264 decoded frames, 15 frames/s) (a) Interpolated frame using FA. (b) Interpolated frame using SMVF. (c) Interpolated frame using MCI+FR. (d) Predicted frame with interpolation errors. (e) Interpolated frame using GIR-MCI. (f) Interpolated frame using ERIR-MCI. (g) Interpolated frame using DB-FMCI

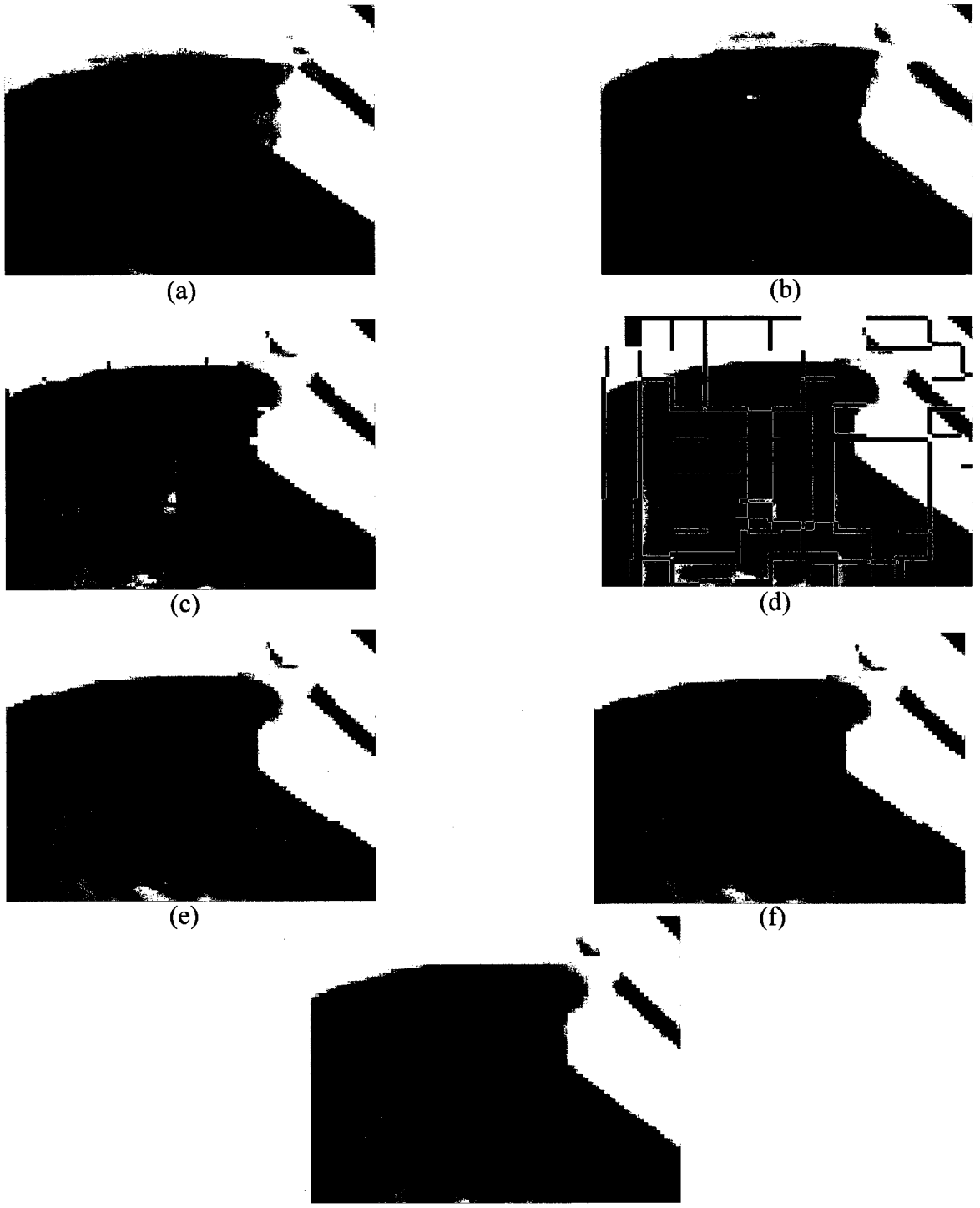


Figure 5.11: Interpolated frames (face portion only) using various schemes corresponding to frame 179 of the *Foreman* sequence. (Simulated with the original frames, 10 frames/s) (a) Interpolated frame using FA. (b) Interpolated frame using SMVF. (c) Interpolated frame using MCI+FR. (d) Predicted frame with interpolation errors. (e) Interpolated frame using GIR-MCI. (f) Interpolated frame using ERIR-MCI. (g) Interpolated frame using DB-FMCI

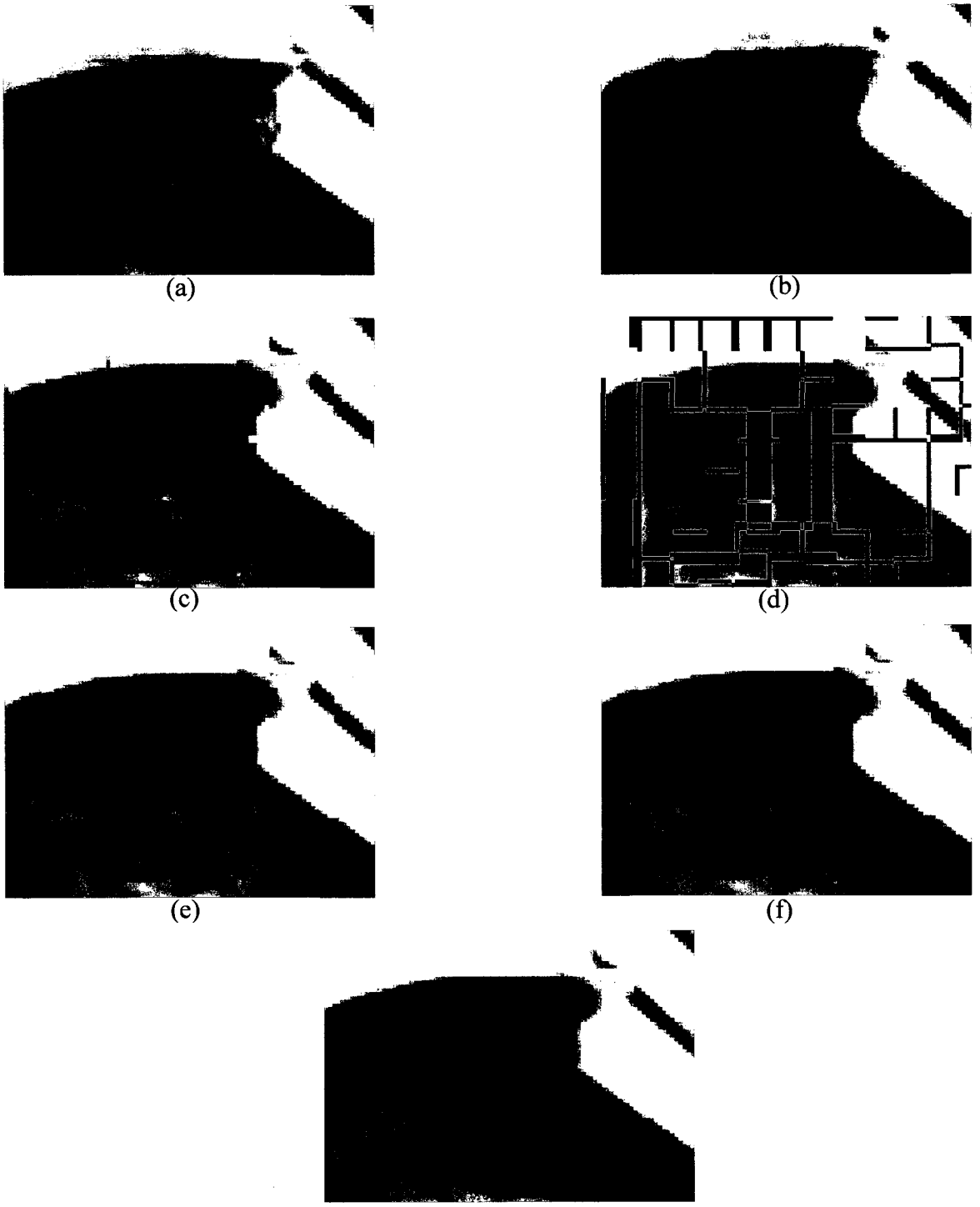


Figure 5.12: Interpolated frames (face portion only) using various schemes corresponding to frame 179 of the *Foreman* sequence. (Implemented in the JM15.0 H.264 decoder, 10 frames/s) (a) Interpolated frame using FA. (b) Interpolated frame using SMVF. (c) Interpolated frame using MCI+FR. (d) Predicted frame with interpolation errors. (e) Interpolated frame using GIR-MCI. (f) Interpolated frame using ERIR-MCI (g) Interpolated frame using DB-FMCI

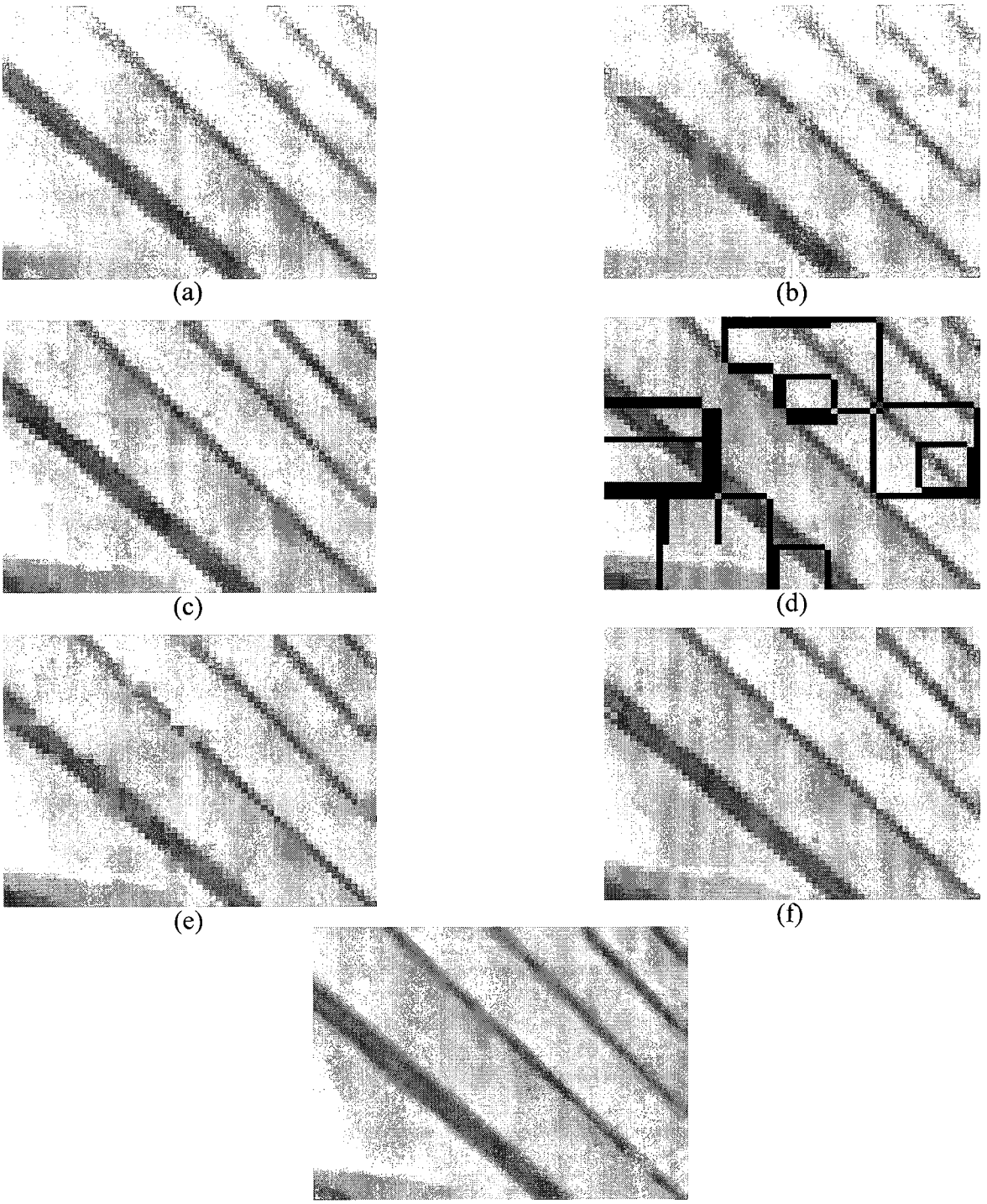


Figure 5.13: Interpolated frames (background part only) using various schemes corresponding to frame 179 of the *Foreman* sequence. (Simulated with the original frames, 10 frames/s) (a) Interpolated frame using FA. (b) Interpolated frame using SMVF. (c) Interpolated frame using MCI+FR. (d) Predicted frame with interpolation errors. (e) Interpolated frame using GIR-MCI. (f) Interpolated frame using ERIR-MCI (g) Interpolated frame using DB-FMCI

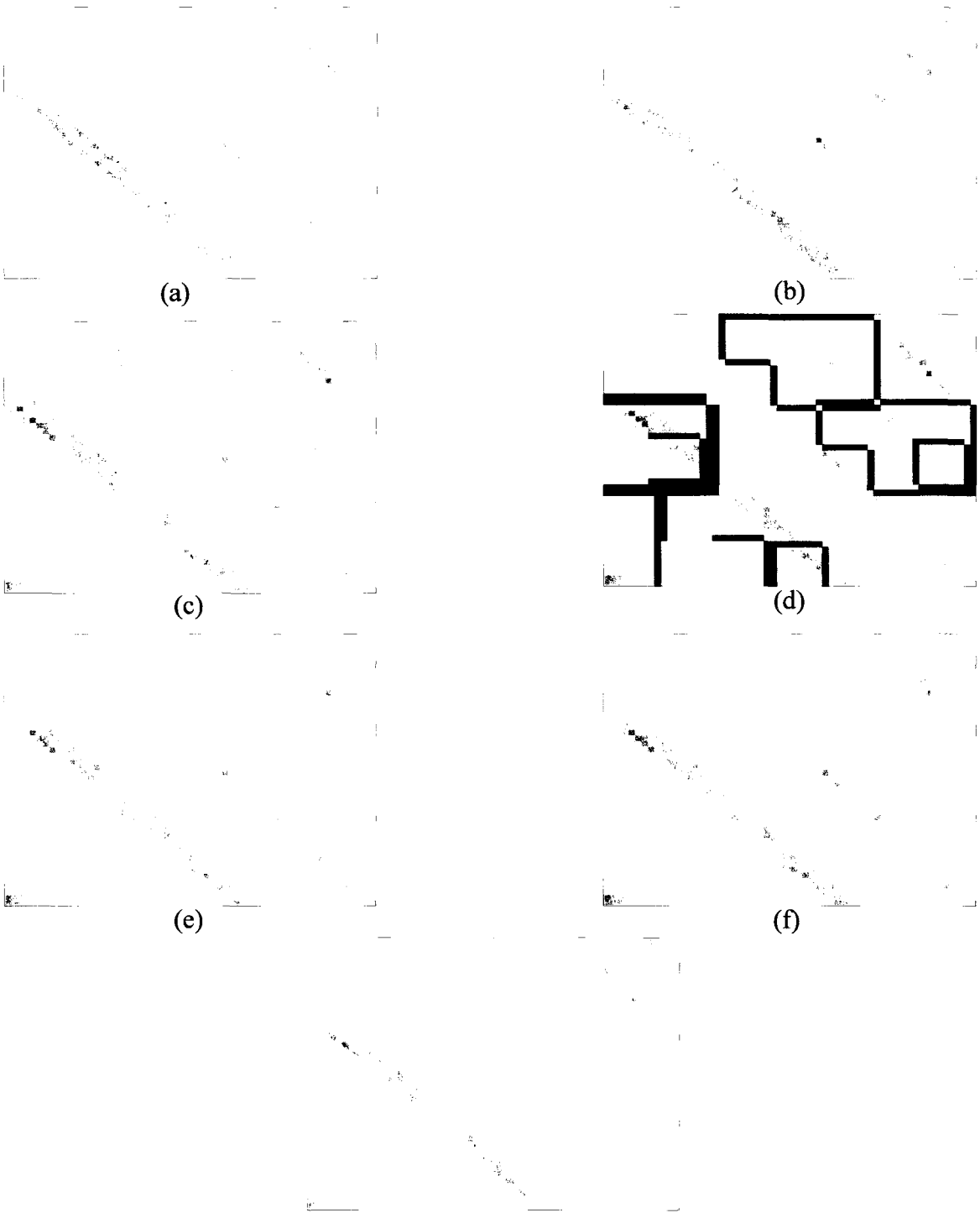


Figure 5.14: Interpolated frames (background part only) using various schemes corresponding to frame 179 of the *Foreman* sequence. (Implemented in the JM15.0 H.264 codec, 10 frames/s) (a) Interpolated frame using FA. (b) Interpolated frame using SMVF. (c) Interpolated frame using MCI+FR. (d) Predicted frame with interpolation errors. (e) Interpolated frame using GIR-MCI. (f) Interpolated frame using ERIR-MCI (g) Interpolated frame using DB-FMCI

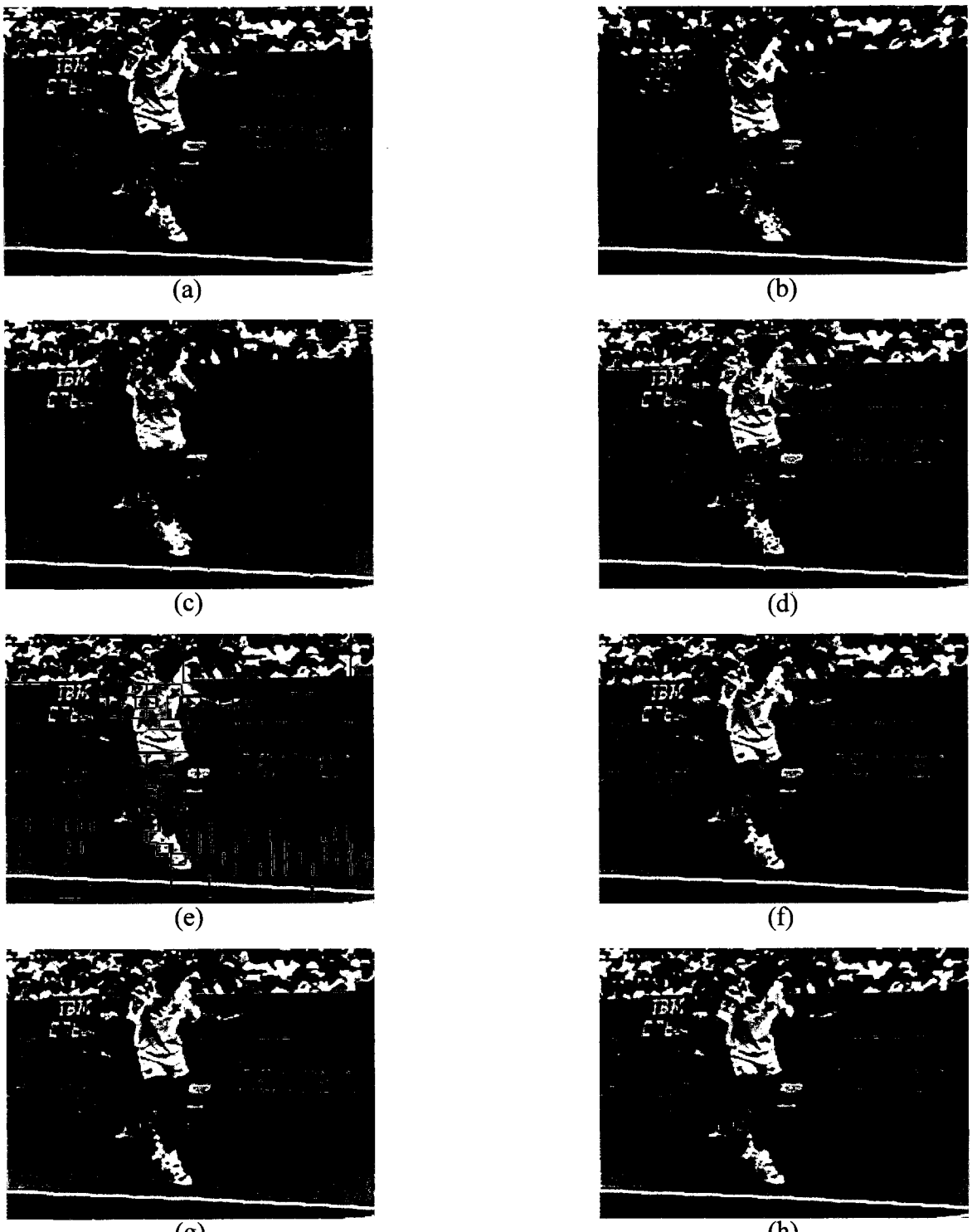


Figure 5.15: Original and interpolated frames (center portion) using various schemes corresponding to frame 21 of the CIF *Stefan* sequence. (Simulated with the original frames, 10frames/s) (a) Original. (b) Interpolated frame using FA. (c) Interpolated frame using SMVF. (d) Interpolated frame using MCI+FR. (e) Predicted frame with interpolation errors. (f) Interpolated frame using GIR-MCI. (g) Interpolated frame using ERIR-MCI (h) Interpolated frame using DB-FMCI

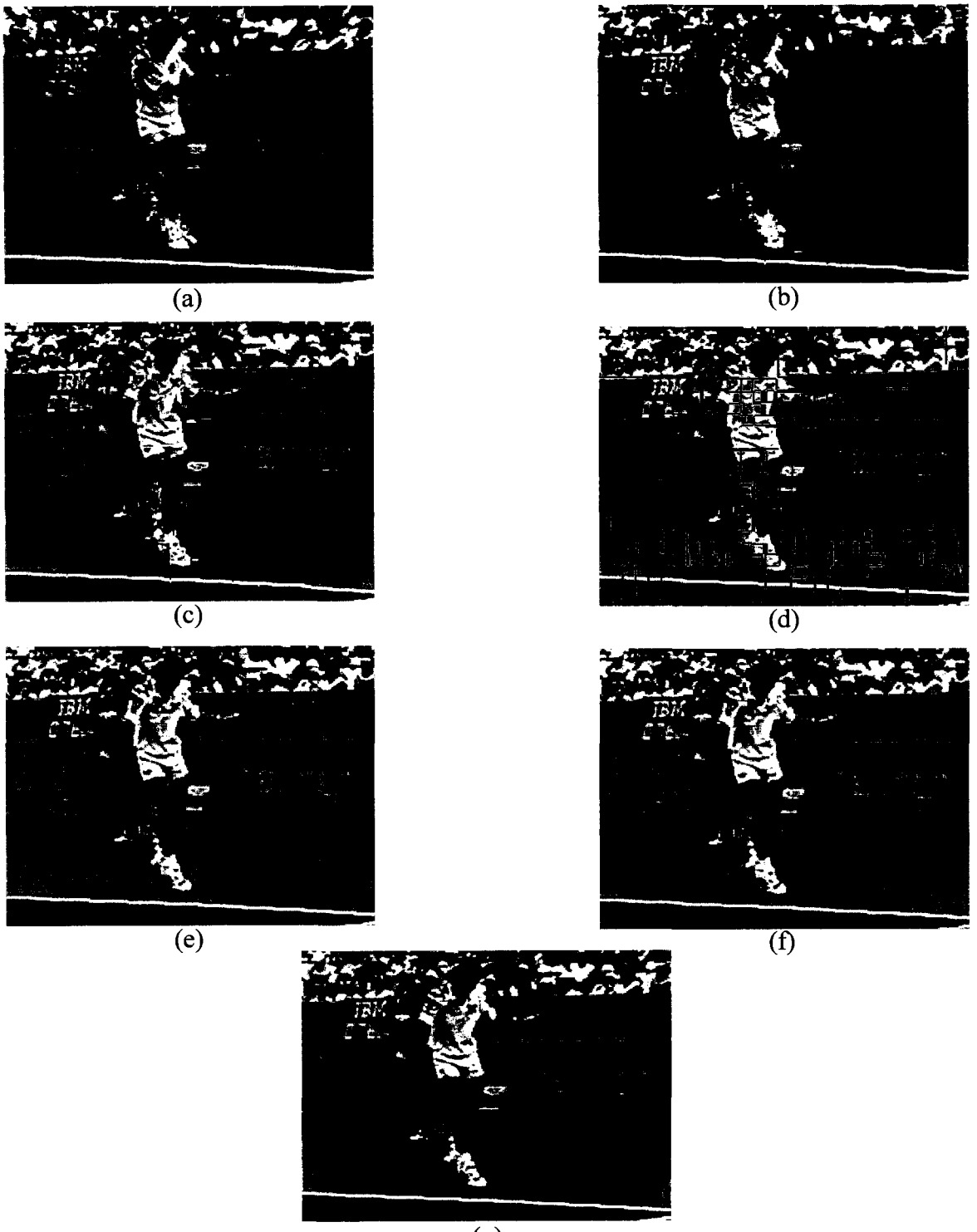


Figure 5.16: Interpolated frames (center portion) using various schemes corresponding to frame 21 of the CIF *Stefan* sequence. (Implemented in the JM15.0 H.264 decoder, 10 frames/s) (a) Interpolated frame using FA. (b) Interpolated frame using SMVF. (c) Interpolated frame using MCI+FR. (d) Predicted frame with interpolation errors. (e) Interpolated frame using GIR-MCI. (f) Interpolated frame using ERIR-MCI (g) Interpolated frame using DB-FMCI

sequences respectively. The results on the center portion of the interpolated frame are presented in Fig. 5.15 and Fig. 5.16 respectively. As is to be expected, the interpolated frames do not look as good as those obtained from the 15 frames/s image sequences irrespective of the scheme employed. However, as in the case of the 15 frames/s image sequences, the proposed GIR-MCI and ERIR-MCI schemes outperform the other ones even in the case of the 10 frames/s image sequences.

Table 5.9 - Table 5.12 show the average number of basic operations needed for each of the schemes implemented in our experiments. The results obtained from both the 15 frames/s and 10 frames/s image sequences are presented. As mentioned earlier, DB-FMCI includes some additional steps, resulting in a high computational load in its implementation. Thus, the computational load of the sophisticated DB-FMCI is much higher than that of SMVF, MCI+FR, and our GIR-MCI and ERIR-MCI. GIR-MCI and ERIR-MCI schemes have a complexity that is lower than that of MCI+FR, and higher than that of SMVF. On average, the number of basic operations in DB-FMCI is about 75 times higher than that of SMVF, with an average improvement of 2.43dB in the PSNR value, while the number of operations of our method is about only 3 times higher than that of SMVF, with an average PSNR improvement of 1.71 dB. Thus, our method is more cost-effective. In fact, the proposed schemes have a much lower computational complexity than that of any other block-based MCI schemes employing pixel classification that introduces the operations required for the detection of scene changes for each pixel, as well as the analysis of motion vector fields at the decoder. The ERIR-MCI scheme provides better subjective and objective qualities than the GIR-MCI does, at the price of a higher computational complexity. Due to the low computational complexity of the Roberts operator, the computational complexity of the ERIR-MCI scheme is still

lower than that of the sophisticated MCI schemes, where a pixel-based motion compensation needs to be implemented. The ERIR-MCI and GIR-MCI schemes provide a tradeoff between performance and computational complexity, according to the requirements of a given application.

Table 5.9: Average number of basic operations in the interpolated frame using various schemes (tested with 15 frames/s sequences, original frames)

Original frames (15 frames/s)	SMVF	MCI+FR	GIR-MCI	ERIR-MCI	DB-FMCI
Mother&Daughter	1,624	27,432	3,795	4,492	91,488
Miss America	1,764	27,612	3,000	3,510	91,728
Suzie	7,287	34,713	28,933	34,230	101,196
Foreman	7,441	34,911	28,407	33,511	101,460
Container	842	102,468	1,854	1,947	307,872
News	2,678	104,848	5,602	5,882	316,032
Coast Guard	11,119	115,789	33,828	35,519	353,544
Stefan	17,248	123,734	55,630	58,412	380,784

Table 5.10: Average number of basic operations in the interpolated frame using various schemes (tested with 15 frames/s sequences, decoded frames)

Decoded frames (15 frames/s)	SMVF	MCI+FR	GIR-MCI	ERIR-MCI	DB-FMCI
Mother&Daughter	1,820	27,684	4,290	5,096	88,704
Miss America	2,002	27,918	3,461	4,018	92,136
Suzie	8,568	36,360	37,944	39,841	103,392
Foreman	8,785	36,639	38,830	40,255	103,764
Container	891	102,531	1,965	2,063	308,088
News	2,865	104,559	5,938	6,235	316,860
Coast Guard	11,829	116,710	34,979	36,728	356,700
Stefan	20,282	127,668	60,770	63,809	394,272

Table 5.11: Average number of basic operations in the interpolated frame using various schemes (tested with 10 frames/s sequences, original frames)

Original frames (10 frames/s)	SMVF	MCI+FR	GIR-MCI	ERIR-MCI	DB-FMCI
Mother&Daughter	1,823	27,653	4,133	4,863	91,620
Miss America	1,980	27,852	3,169	3,697	91,872
Suzie	8,183	35,709	30,547	36,002	101,796
Foreman	8,355	35,927	31,193	35,649	102,072
Container	1,080	102,776	2,090	2,195	308,928
News	3,024	105,296	6,594	6,924	317,568
Coast Guard	14,129	119,692	36,854	38,697	366,924
Stefan	21,743	129,562	64,270	67,484	400,764

Table 5.12: Average number of basic operations in the interpolated frame using various schemes (tested with 10 frames/s sequences, decoded frames)

Decoded frames (10 frames/s)	SMVF	MCI+FR	GIR-MCI	ERIR-MCI	DB-FMCI
Mother&Daughter	2,033	27,919	4,610	5,424	91,956
Miss America	2,235	28,175	3,580	4,325	92,280
Suzie	9,578	37,476	40,852	42,129	104,028
Foreman	9,818	37,780	40,578	41,887	104,412
Container	1,158	102,878	2,215	2,326	309,276
News	3,205	106,124	6,990	7,340	318,372
Coast Guard	15,088	120,934	38,926	40,872	371,184
Stefan	24,114	132,635	68,945	72,392	411,300

5.4 Summary

This chapter has presented error concealment-based motion-compensated interpolation (MCI) schemes that exploit the block-based motion vector field available at the decoder. The generic iterative refinement (GIR) module derived through the use of the finite element model has been employed to effectively conceal the interpolation errors caused by the unfilled and overlapped pixels in the compensated frames. It provides a fine texture in relatively smooth areas for the interpolated frames. Based on the GIR, an edge reuniting iterative refinement (ERIR) module has been developed, and this module has the capability of maintaining the pixel continuity along the edge, and hence, works well in sharp edge regions for the interpolated frames. The GIR module can itself be used as an MCI scheme, while the ERIR-MCI scheme is composed of both the GIR and ERIR modules, along with a module selector. No pixel classification or motion estimation is needed for either of these schemes, thus substantially reducing the computational complexity. Two experiments have been carried out to evaluate the performance of the schemes developed. The first experiment deals with interpolation using the original images from sub-sampled test sequences, and the second involves the implementation of the proposed schemes with the JM15.0 H.264 decoded frames. Simulation results from these experiments have shown that the proposed schemes result in interpolated frames with good visual qualities, at a low computational cost. When compared to the GIR-MCI scheme, the ERIR-MCI scheme achieves even better subjective and objective qualities at the price of a higher computational complexity. The ERIR-MCI and GIR-MCI schemes

provide a tradeoff between the performance and computational complexity, according to the requirements of a given application.

Chapter 6

Conclusion and Future Study

6.1 Concluding Remarks

Spatial resolution enhancement comprises reconstructing a high resolution image from a low resolution image, whereas temporal resolution enhancement of encoded video aims at interpolating the skipped frames, making use of two successively received frames. In this thesis, new schemes for edge-directed spatial resolution enhancement and block-based MCI schemes for temporal resolution enhancement of encoded video sequences, have been proposed.

Even though a number of edge-directed interpolation schemes have been developed, they do not take into account the neighboring unknown pixels when interpolating a pixel for an up-scaled image. Therefore, these schemes do not give a very natural looking reconstructed image, especially in areas with fine textures. Also, due to the fact that existing edge-directed interpolation schemes are commonly designed for a magnification factor of two, these schemes can only be applied when the magnification factor is an integer power of two. Consequently, these schemes need the use of some conventional interpolation methods when magnification factor is not a power of two. Moreover, since the existing schemes interpolate unknown pixels based only on the estimation made with known pixels, these schemes require more than one step to

complete the interpolation process even when an image is up-scaled by a factor of two. Obviously, such a scheme needs to be applied n times to up-scale an image by a factor of 2^n, n being an integer greater than one.

There are two major problems associated with block-based MCI scheme for temporal resolution enhancement of encoded video sequences, there are two major problems related to this topic, namely, as to how to deal with overlapped pixels in the interpolated frames, and as to how to handle the holes left in the interpolated frames. Existing methods employ the averaging technique to handle the overlapped pixels, while techniques such as the pixel averaging or repetition are used to fill the holes. These simple techniques result in interpolated frames that suffer from blur and stripe effects.

To solve the above problems, a new image interpolation model has been developed in this thesis. The finite element method, which offers the simplicity of piecewise approximation of a function given its values at discrete points, has been chosen and utilized to design such an image interpolation model. Based on this model, an image interpolation technique, which takes into account not only the neighboring known pixels, but also the neighboring pixels with unknown values so as to provide a spatial continuity between the unknown pixels as well, has been proposed. It is very general in that it can be used to interpolate a collection of unknown pixels with an arbitrary shape. It should be noted that, FEM, a powerful numerical mathematical tool that has been employed for obtaining numerical solution to a wide variety of engineering problems, has been applied in this thesis for the first time in the spatial and temporal resolution enhancement of images.

Based on this generalized image interpolation model, an edge-preserving iterative refinement (EPIR) scheme has been developed for the application of spatial resolution enhancement of images. This scheme is not restricted to the neighboring pixels with known values; it also utilizes the neighboring pixels with unknown values through an iterative refinement technique. Hence, the proposed scheme is capable of reconstructing natural-looking images in areas with fine textures. It has been shown, by using the Roberts operator as the gradient estimator, that the edge-preserving iterative refinement process provides a smooth variation along a dominant edge in the up-scaled image. Through experiments, it has been shown that the proposed scheme results in up-scaled images with better subjective and objective qualities than that provided by the other edge-directed interpolation schemes, including EDIM, one of the best methods in its category. Further, it has a much lower computational complexity than EDIM. The scheme is also characterized by being capable of up-scaling an image by an arbitrary magnification factor u that is not restricted to be an integer power of two. Furthermore, the proposed scheme needs to be applied only once irrespective of the value of the magnification factor.

As to the block-based MCI for temporal resolution enhancement of encoded video sequences, an error concealment-based MCI scheme that utilizes the block-based motion vector field available at the decoder has been developed. This scheme consists of a module selector, a generic iterative refinement (GIR) module and an edge-reuniting iterative refinement (ERIR) module. The module selector is utilized to determine as to which specific interpolation module, GIR or ERIR, is to be used for the concealment of the interpolation errors. The GIR module has been developed based on the generalized

image interpolation model, and is capable of effectively concealing the interpolation errors caused by the overlapped pixels and unfilled pixels in the compensated frames. It provides a fine texture in homogeneous areas for the interpolated frames. The GIR module can itself be used as an independent MCI scheme. In order to handle the case when dominant edges exist in the interpolated frame, an edge reuniting iterative refinement (ERIR) module has also been developed, and this module is capable of maintaining the pixel continuity along the edge. Therefore, it can perform well in dominant edge regions in the interpolated frames. An MCI scheme, termed the ERIR-MCI scheme consisting of both the GIR and ERIR modules, along with a module selector has also been proposed to handle the cases where dominant edges are present in the interpolated frames. These two error concealment-based MCI schemes are characterized by the fact that no pixel classification or motion estimation is needed for either of these schemes, thus substantially reducing the computational complexity. Extensive experiments have been carried out to evaluate the performance of the two schemes. The first set of experiments deal with interpolation using the original images from sub-sampled test sequences, whereas the second set involves the implementations of the proposed schemes with the JM15.0 H.264 decoded frames. Results from these experiments have shown that the proposed schemes result in interpolated frames with good visual qualities, at a low computational cost. The ERIR-MCI provides even better subjective and objective qualities than the GIR-MCI scheme, at the cost of a higher computational complexity. The ERIR-MCI and GIR-MCI schemes can be selected for use according to the requirements of a given application.

6.2 Suggestions for Future Investigation

In this thesis, we have employed the gradient estimator and directional classification of pixels in order to detect the dominant edge inside an image block. As pointed out in Chapter 4, it is normal that an image block contains more than one discernible edge. If an interpolation method along only a given orientation is carried out, it may generate a false pixel structure. Hence, it would be worthwhile to undertake a study involving interpolation along multiple edges within a local block to find a solution to this problem.

Most MCI schemes have been developed based on a constant-speed motion model, which is a reasonable assumption for video sequences without very fast motions. However, for an encoder operating at a low bitrate using the frame dropping technique, the motion between the coded frames becomes larger than that between two consecutive frames at the normal frame rate, and therefore, a more comprehensive motion model could be explored to obtain a smoother motion trajectory.

References

[1] R. G. Keys, "Cubic convolution interpolation for digital image processing," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 29, pp.1153–1160, Dec. 1981.

[2] J. A.Parker, R.V.Kenyon and D.E.Troxel, "Comparison of Interpolating methods for Image Resampling," *IEEE Trans. on Image Processing*, vol. MI-2, no.l, March 1983;

[3] P. Thévenaz, T. Blu, and M. Unser, "Interpolation revisited," *IEEE Trans. Med. Imag.*, vol. 19, pp. 739–758, July 2000.

[4] K. Ratakonda and N. Ahuja, "POCS based adaptive image magnification," in *Proc. IEEE Int. Conf. Image Processing*, vol. 3, pp.203-207, 1998.

[5] D. Calle and A. Montanvert, "Superresolution inducing of an image," in *Proc. IEEE Int. Conf. Image Processing*, vol. 3, pp. 232-235, 1998.

[6] T. Blu, P. Thévenaz, and M. Unser, "MOMS: Maximal-order interpolation of minimal support," *IEEE Trans. Image Processing,* vol. 10, pp.1069–1080, July 2001.

[7] W. K. Carey, D. B. Chuang, and S. S. Hemami, "Regularity-preserving image interpolation," *IEEE Trans. Image Process.*, vol. 8, no. 9, pp. 1293–1297, Sep. 1999.

[8] Y. Zhu, S. C. Schwartz, and M. T. Orchard, "Wavelet domain image interpolation via statistical estimation," in *Proc. Int. Conf. Image Processing*, vol. 3, pp. 840–843, Sep. 2001.

[9] D. D. Muresan and T. W. Parks, "Prediction of image detail," in *Proc. Int. Conf. Image Processing*, vol. 2, pp. 323–326, Sep. 2000.

[10] D. A. Florencio and R. W. Schafer, "Post-sampling aliasing control for natural images," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing*, vol. 2, pp. 893-896, 1995.

[11] F. Fekri, R. M. Mersereau, and R.W. Schafer, "A generalized interpolative VQ method for jointly optimal quantization and interpolation of images," in Proc. *IEEE Int. Conf. Acoustics, Speech, Signal Processing*, vol. 5, pp. 2657-2660, 1998.

[12] J. E. Adams Jr, "Interactions between color plane interpolation and other image processing functions in electronic photography," *Proc. SPIE*, vol. 2416, pp. 144–151, 1995.

[13] K. Jensen and D. Anastassiou, "Subpixel edge localization and the interpolation of still images," *IEEE Trans. on Image Processing*, vol. 4, pp.285–295, Mar. 1995.

[14] V. R. Algazi, G. E. Ford, and R. Potharlanka, "Directional interpolation of images based on visual properties and rank order filtering," *in Proc.IEEE Int. Conf. Acoustics, Speech, Signal Processing*, vol. 4, pp.3005-3008, 1991.

[15] J.Allebach and P.W.Wong, "Edge-directed Interpolation," *Proceeding of ICIP'1996*, pp.707-710, 1996.

[16] S. Carrato and L. Tenze, "A high quality 2× image interpolator," *IEEE Signal Process. Lett.*, vol. 7, no. 6, pp. 132–135, Jun. 2000.

[17] L. Zhang and X. Wu, "Image interpolation via directional filtering and data fusion," IEEE Trans. Image Process., vol. 15, no. 8, pp.2226–2238, Aug. 2006.

[18] X.Li and M.T.Orchard, "New Edge Directed Interpolation," *IEEE Transactions on Image Processing*, vol.10, no. 10, pp.1521-1527, 2001.

[19] Xiaolin Wu and Xiangjun Zhang, "Image interpolation using texture orientation map and kernel Fisher discriminant," in *Proc. Int. Conf. Image Processing*, vol.1, pp. 49-52, Sept. 2005.

[20] Q. Wang and R. Ward, "A new edge-directed image expansion scheme," in *Proc. Int. Conf. Image Processing*, vol.3 pp.899 – 902, 2000.

[21] Xiangjun Zhang and Xiaolin Wu, "Image Interpolation by Adaptive 2-D Autoregressive Modeling and Soft-Decision Estimation," *IEEE Transactions on Image Processing*, vol. 17, no. 6, pp. 887-896, June 2008.

[22] R.Thoma and M.Bierling, "Motion compensated interpolation considering covered and uncovered background," *Signal Processing:Image Communication*, vol.1, pp.1191 –212, 1989.

[23] C. Cafforio, F. Rocca, and S. Tubaro, "Motion compensated image interpolation," *IEEE Trans. Commun*, Vol. 38, No. 2, pp. 215–222, 1990.

[24] J.H. Kim and C.M. Kyung, "Fast frame-to-frame interpolation technique for scenes containing moving objects," *Electron.Lett*, Vol. 27, No. 20, pp. 1788–1790, 1991.

[25] L. Zhao and Z. Zhou, "A new algorithm for motion compensated frame interpolation," in *Proceedings of the1993 IEEE International Symposium on Circuits and Systems (ISCAS '93)*, Chicago, USA, Vol. 1, pp. 9-12, May 1993.

[26] R. Castagno, P. Haavisto, and G. Ramponi, "A method for motion adaptive frame rate up-conversion," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol.6, No. 5, pp.436-446, 1996.

[27] H. Yamaguchi, T. Sugi and K, Kinuhata, "Movement-compensated frame-frequency conversion of television signals," *IEEE Transactions on Communication*, vol. 35, no. 10, pp. 1069-1082, 1987.

[28] P. Robert, "Motion compensating interpolation considering occluding, appearing, and disappearing areas," *Signal Processing of HDTV*, III, pp. 329-341, 1992.

[29] H. Blume, "Nonlinear vector error tolerant interpolation of intermediate video images by weighted medians," *Signal Processing: Image Communication*, vol.18 pp.851 –868, 1999.

[30] F. Moscheni, F. Dufaux, and M. Kunt, "A new two-stage global/local motion estimation based on a back-ground/foreground segmentation," in *Proc. ICASSP-95*, vol. 4, pp. 2261-2264, 1995.

[31] Mark A. Robertson and Robert L. Stevenson, "Temporal resolution enhancement in compressed video sequences," *EURASIP Journal on Applied Signal Processing: Special Issue on Nonlinear Signal Processing*, pp.230-238, 2001

[32] R. Castagno, P. Haavisto, and G. Ramponi, "A method for motion adaptive frame rate upconversion," *IEEE Trans. on CSVT*, vol. 6, no.5, pp.436–446, Oct. 1996.

[33] C.-K.Wong and O. C. Au, "Fast motion compensated temporal interpolation for video," in *Proc. of SPIE Visual Comm. and Image Processing*, pages 1108–1118, Taipei, Taiwan, May 24–26 1995.

[34] F. Moscheni, F. Dufaux, and M. Kunt, "A new two-stage global/local motion estimation based on a back-ground/foreground segmentation," in *Proc. ICASSP95*, vol. 4, pp. 2261-2264, 1995.

[35] Long Zhao and Zheng Zhou, "A new algorithm for motion-compensated frame interpolation," in *Proc. Int. Conf. Image Processing*, pp.9-12, 1993.

[36] K. Kawaguchi and S. K. Mitra, "Frame rate up-conversion considering multiple motion," in *Proc. Int. Conf. Image Processing*, pp.727-730, 1997.

[37] S.-C. Han and J. W. Woods, "Frame-rate up-conversion using transmitted motion and segmentation fields for very low bit-rate video coding," in *Proc. Int. Conf. Image Processing*, vol. 1, pp. 747-750, Oct. 1997.

[38] T.Y. Kuo, J.W. Kim, and C.-C. Jay Kuo, "Motion-compensated frame interpolation scheme for H.263 Codec," *in Proc. of ISCAS 99*, vol. 4, pp.491-494, May 1999.

[39] Shan Liu, J.W. Kim, and C.-C. Jay Kuo "MCI-embedded motion compensated prediction for quality enhancement of flame interpolation," in *SPIE Proceeding of International Symposium on Voice, Video, and Data Communication, Multimedia systems and applications III*, 2000.

[40] Y.-K. Chen, A. Vetro, H. Sun, and S. Y. Kung, "Frame-rate up-conversion using transmitted true motion vectors," *in Proc. IEEE Second Workshop on Multimedia Signal Processing*, pp. 622-627, Dec. 1998.

[41] Seung-Chul Yoon and Narendra Ahuja, "Frame Interpolation Using Transmitted Block-Based Motion Vectors," in *Proc. of ICIP'2001*, pp. 856-859, 2001.

[42] T.Y. Kuo and C.C.J. Kuo, "Motion-compensated interpolation for low-bit-rate video quality enhancement," *SPIE Conference on Application of Digital Image Processing XXI*, San Diego, California, vol.3460, pp.277-288, 1998.

[43] J.K. Su and R.M. Mersereau, "Motion-compensated interpolation of untransmitted frames in compressed video," *Conference Record of the Thirtieth Asilomar Conference on Signals, Systems and Computers*, Pacific Grove, USA, pp.100 –104,1996.

[44] T. Liu, K.T. Lo, J. Feng, and X. Zhang, "Frame interpolation scheme using inertia motion prediction," Signal Processing: Image Communication, vol. 18, pp.221 – 229, 2003.

[45] A. Pelagotti and G. de Haan, "A new algorithm for high quality video format conversion," in *Proc. of ICIP'2001*, pp.375-378, 2001.

[46] Chung-Tao Chu, Dimitris Anastassiou and Shih-Fu Chang, "Hierarchical global motion estimation/compensation in low bitrate video coding," in *Proc. of IEEE International Symposium on Circuits and Systems*, June 1997.

[47] Yan Wu, M.N.S Swamy, and M.O. Ahmad, "Concealment of interpolation errors for low bit-rate motion-compensated interpolation," in *Proc. of ICIP'2004*, pp.981-984, 2004.

[48] Yan Wu, M.N.S Swamy, and M.O. Ahmad, "Error concealment for motion compensated interpolation," *IET Image Processing*, in Print, 2010.

[49] Yan Wu, M.O. Ahmad, and M.N.S Swamy, "Iterative refinement interpolation for image up-scaling," Under review.

[50] Yan Wu, M.O. Ahmad, and M.N.S Swamy, "A two-mode interpolation error concealment MCI scheme," *IEEE-NEWCAS Conference*, pp. 43- 46, June 2005

[51] W. M. Lam, A. R. Reibman, and B. Liu, "Recovery of lost or erroneously received motion vectors," in *Proc. of ICASSP'93*, vol. V, pp. 417–420, 1993.

[52] Y. Wang, Q. F. Zhu, and L. Shaw, "Maximally smooth image recovery in transform coding," *IEEE Trans. Commun.*, vol. 41, pp. 1544–1551, Oct. 1993.

[53] H. Sun and W. Kwok, "Concealment of damaged block transform coded images using projection onto convex sets," *IEEE Trans. Image Processing*, vol. 4, pp. 470–477, Apr. 1995.

[54] W. Kwok and H. Sun, "Multi-directional interpolation for spatial error concealment," *IEEE Trans. Consumer Electron.*, vol. 39, pp. 455–460, Aug. 1993.

[55] P. Salama, N. Shroff, E. J. Coyle, and E. J. Delp, "Error concealment in encoded video streams," in *Signal Recovery Techniques for Image and Video Compression and Transmission*, Eds. Boston, MA: Kluwer Academic, 1998.

[56] R. Talluri, "Error resilient video coding in the MPEG-4 standard," *IEEE Commun. Mag.*, vol. 26, pp. 112–119, June 1998.

[57] S. Shirani, F. Kossentini, and R.Ward, "Reconstruction of motion vector missing macroblocks in H.263 encoded video transmission over lossy networks," in *Proc. of ICIP*, vol. III, pp.487–491, Oct. 1998.

[58] Y. Wang, et al., "Error resilient video coding techniques," *IEEE Signal Processing Magazine*, PP. 61-82, July 2000.

[59] Y. Yang, N. P. Galatsanos, and A. K. Katsaggelos, "Regularized reconstruction to reduce blocking artifacts of block discrete cosine transform compressed images," *IEEE Trans. on CSVT*, vol.3, no.6, pp.421–432, Dec.1993.

[60] Y. Yang, N. P. Galatsanos, and A. K. Katsaggelos, "Projection-based spatially adaptive reconstruction of block-transform compressed images," *IEEE Trans. on Image Processing*, vol.4, no.7, pp.896–908, July 1995.

[61] J. Mateos, C. Ilia, B. Jim´enez, R. Molina, and A. K.Katsaggelos, "Reduction of blocking artifacts in block transformed compressed color images," in *Proc. of International Conference on Image Processing*, vol.1, pp. 401–405, Oct. 4–7 1998.

[62] P. Salama, N. Shroff, and E. J. Delp, "A Bayesian approach to error concealment in encoded video streams," in *Proc. of International Conference on Image Processing*, vol.2, pp. 49–52, 1996.

[63] O.C. Zienkiewicz and R.L. Taylor, *The Finite Element Method*, 5th edition, Oxford; Boston: Butterworth-Heinemann, 2000.

[64] K. Huebner and E.A. Thornton, *The Finite Element Method for Engineers*, New York: Wiley, 1982.

[65] Ronald L. Huston and Chris E. Passerello, *Finite Element Methods: An Introduction*, New York : M. Dekker, 1984.

[66] H.R. Schwarz, *Finite Element Methods*, London : Academic Press, 1988.

[67] I. Pitas, *Digital Image Processing Algorithms and Applications*, New York: John Wiley & Sons, 2000.

[68] Bernd Jähne, *Digital image processing: concepts, algorithms, and scientific applications*, 4th edition, Berlin; New York : Springer, 1997.

[69] http://iphome.hhi.de/suehring/tml/download , accessed July 2009.