NOTE TO USERS

Université d'Ottawa · University of Ottawa

Université d'Ottawa • University of Ottawa

FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTDOCTORALES

FACULTY OF GRADUATE AND POSTDOCTORAL STUDIES

Jeremy Alain RANGER

AUTEUR DE LA THÈSE - AUTHOR OF THESIS

M. A. Sc. (Electrical Engineering)

GRADE - DEGREE

Department of Electrical Engineering

FACULTÉ, ÉCOLE, DÉPARTEMENT - FACULTY, SCHOOL, DEPARTMENT

TITRE DE LA THÈSE - TITLE OF THE THESIS

Adaptative Image Magnification Using Edge-Directed and Statistical Methods

É. Dubois

DIRECTEUR DE LA THÈSE - THESIS SUPERVISOR

CO-DIRECTEUR DE LA THÈSE - THESIS CO-SUPERVISOR

EXAMINATEURS DE LA THÈSE - THESIS EXAMINERS

A. Adler

R. Goubran

J.-M. De Koninck, Ph.D.

LE DOYEN DE LA FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTDOCTORALES

DEAN OF THE FACULTY OF GRADUATE AND POSTDOCTORAL STUDIES

Adaptive Image Magnification Using Edge-Directed and Statistical Methods

Jeremy Ranger

Ottawa-Carleton Institute for Electrical and Computer Engineering
School of Information Technology and Engineering (SITE)
University of Ottawa
Ottawa, Ontario, Canada

June 2004

A Thesis submitted to the Faculty of Graduate and Postdoctoral Studies
in partial fulfillment of the requirements for the degree of
Master of Applied Science, Electrical Engineering

Your file Votre référence
ISBN: 0-494-01588-8
Our file Notre référence
ISBN: 0-494-01588-8

Abstract

This thesis provides a comparison of two adaptive image magnification algorithms selected from the literature. Following implementation and experimentation with the algorithms, a number of improvements are proposed.

The first selected algorithm takes an edge-directed approach by using an estimation of the edge map of the high-resolution image to guide the interpolation process. It was found that this algorithm suffered from certain inaccuracies in the edge detection stage. The proposed improvements focus on methods for increasing the accuracy of edge detection.

The second selected algorithm takes a statistical approach by modelling the high-resolution image as a Gibbs-Markov random field and solving with the maximum *a posteriori* estimation technique. It was found that this algorithm suffered from blurring caused by the general way in which the clique potentials are applied to every sample. The proposed improvements introduce a set of weights to prevent smoothing across discontinuities.

The two selected algorithms are compared to the enhanced versions to demonstrate the merit of the proposed improvements. Results have shown significant improvements in the quality of the magnified test images. In particular, blurring was reduced and edge sharpness was enhanced.

Acknowledgments

First and foremost, I would like to thank my parents for their support. Their encouragement to continue my studies has led me to the completion of this thesis.

I would like to thank my thesis supervisor, Eric Dubois, for providing me with the opportunity to conduct this work. His patience and guidance have helped me carry this through to the end.

I would like to thank Hussein Aly for his help in getting my research kick-started. His counsel proved invaluable in clarifying some of the more esoteric aspects of the selected algorithms.

I would also like to thank the Royal Canadian Mounted Police for funding this work under contract no. M9010-3-0764.

Contents

List of Figures

List of Tables

Notation

The following is a list of symbols that will be used throughout the document.

Symbol	Description
HR	High-resolution
LR	Low-resolution
PDE	Partial-differential equation
COSO	Center-On-Surround-Off filter
Unit Cell	Area between any four samples on a regularly spaced rectangular grid
M_x	Horizontal magnification factor
M_y	Vertical magnification factor
X_{LR}	Horizontal dimension of the low-resolution image
Y_{LR}	Vertical dimension of the low-resolution image
X_{HR}	Horizontal dimension of the high-resolution image
Y_{HR}	Vertical dimension of the high-resolution image
$\mathbf{v}, \mathbf{M}$	Bold face symbols represent a vector or matrix
$\mathbf{v}^t, \mathbf{M}^t$	Transpose of a vector or matrix
$M \times N$	Dimensions of a matrix with M rows and N columns
α	Loop gain parameter used in the original and improved edge-directed algorithms
S	Maximum slope of the edge model used in the improved edge-directed algorithm
λ	Regularization parameter applied to the data fidelity term used in the statistical algorithm

T	Threshold parameter applied to the smoothing term used in the statistical algorithm
$\mathbf{D}$	Decimation operator used in the original and improved statistical algorithms
$\|\| \cdot \|\|$	Euclidean norm
$*$	Convolution operator
$\cdot$	Inner (dot) product of a vector or matrix multiplication
λ_1	Weight applied to regions without discontinuities used in the improved statistical algorithm
λ_2	Weight applied to regions with discontinuities used in the improved statistical algorithm
$f[x, y]$	Two-dimensional discrete-space signal
$p(\cdot)$	Probability density function
$\lfloor x \rfloor$	Largest integer less than x

Chapter 1

Introduction

Image magnification is among the fundamental image processing operations. Applications are varied and range from the specialized to the mundane. In medical imaging, magnification can serve to improve the chances of diagnosing problems by highlighting any possible aberrations. Enhancing image details can also be useful for the purposes of identification, whether for improving the quality of an image interpreted by a biometric recognition system or trying to get a clearer view of the perpetrator of some crime. In entertainment, magnification can be used to resize a video frame to fit the field of view of a projection device, which may help to reduce blurring. Finally, the most obvious application of image magnification is to simply allow one to enjoy a larger version of a favorite image obtained from any commercially available digital imaging device such as a camera, camcorder or scanner.

Conventional image magnification techniques use up-sampling by zero-insertion followed by linear filtering to interpolate the high-resolution samples. The main drawback of this approach is that the frequency content of the high-resolution image is the same as the low-resolution image. This is due to the fact that linear techniques are incapable of introducing new information into the image. The lack of new high frequency content results in a variety of undesirable image artifacts such as blocking, staircase edges and blurring.

Adaptive image magnification techniques attempt to reconstruct the high frequency content of the high-resolution image by making use of information in the low-resolution image. With this approach, high frequencies, which promote sharp edges and minute details, are integrated into the high-resolution image based on the specific content of a particular low-resolution image.

1.1 Motivation

The problem of image magnification has been pursued in a variety of ways in the literature. However, despite the wealth of existing work in the field, there are very few commercial software packages that offer adaptive techniques. Most current digital imaging software packages such as Adobe PhotoShop, as well as software included with digital cameras and scanners, implement the conventional linear magnification algorithms. Currently available adaptive techniques are provided in certain highly specialized applications, such as the S-Spline package found at http://www.shortcut.nl [1]. However, these applications have only recently been brought to market and have yet to be included in the popular software products.

The deficiency of adaptive techniques in commercial applications is due to the difficulty involved in implementing and using adaptive algorithms, whose behavior are inherently less predictable than their linear counterparts. Lack of predictability results from the use of tuning parameters, which provide control over some aspect of an algorithm's performance. If not chosen correctly, the tuning parameters can significantly affect the magnified result. By contrast, linear techniques are simple and will always function in a predictable fashion. At the time of this research, there were no adaptive magnification algorithms that provided an automatic means of determining the required parameters such that results are as consistent as linear methods.

The current trends in research show that magnification algorithms are moving towards the more sophisticated techniques offered by adaptive methods. Although linear tech-

[1] Valid at the time of this writing, May 2004.

niques have shown the ability to generate adequate results at low magnification factors, their deficiencies become blatantly apparent as magnification increases. These deficiencies are seen by the introduction of blocking artifacts, blurring and jagged edges, to name but a few. Adaptive techniques are generally targeted to correct such deficiencies, at the expense of requiring tuning parameters.

The primary hurdle in bringing adaptive image magnification algorithms to practical applications is in making the decision of which algorithm to use. Obviously, the best algorithm or a selection of equivalent algorithms should be chosen. However, making such a selection would require implementing and comparing the large variety of currently available algorithms from the literature, since a comprehensive comparison of them has yet to be performed. This solution is not viable due to the development costs needed to produce such a comparison. For this reason, adaptive techniques are limited to highly specialized applications used by research and professional groups that require the utmost quality in image magnification.

1.2 Description of the Thesis Research

Although this work is by no means intended to be a comprehensive comparison of adaptive magnification algorithms, the following document provides a comparison of two adaptive image magnification algorithms selected from the literature. Based on experimentation with the two algorithms, certain improvements were proposed.

This work has been performed in four phases: i) literature survey, ii) implementation, iii) improvements, and iv) comparison.

The first phase required searching the current literature on the topic of adaptive image magnification and extracting the two most promising algorithms for implementation and comparison.

The second phase involved implementing the algorithms as described by the authors of the selected literature. Following implementation, magnification tests were performed

on a variety of images. The testing helped to highlight areas for possible improvement. Implementation of the algorithms was executed using the Matlab scripting language.

The third phase was based on experience gained during the implementation phase. Modifications were made in an attempt to improve on the quality of the magnified images resulting from the original algorithms. The proposed improvements attempted to correct any observed problems with the implementation of each algorithm.

The fourth phase involved a comparison of the quality of magnified test images using the original and improved algorithms. The criteria used for determining the quality of magnified images was purely subjective. This approach was chosen due to the inability of numerical measures to quantify the subjective aspects of human visual perception. The reasoning behind this choice is that an adaptive magnification algorithm must incorporate new content into an image based on currently available information. In practical situations, a high-resolution version of the original image is not given. Obviously, there would be no need for magnification if a high-resolution version were readily available. Therefore, any new content added to an image should be evaluated with respect to magnification with equivalent algorithms. This would allow for comparing the merits of each respective algorithm.

The first selected algorithm, [16], takes an edge-directed approach to image magnification. Edge-directed algorithms focus on using the edges of the low-resolution image to estimate the edges in the high-resolution image. The low-resolution image is then interpolated such that blurring across the estimated edges in the high-resolution image is avoided. This allows edges to remain sharp. It was found that this algorithm suffered from certain inaccuracies in the edge detection stage. The proposed improvements attempt to correct these inaccuracies by choosing a more sensitive second-order operator for the edge detection and introducing a new strategy for estimating the path taken by the edges. Also, the edge-directed rendering stage of the algorithm has been simplified.

The second selected algorithm, [29], takes a statistical approach to image magnification. Statistical algorithms focus on estimating the samples of the high-resolution image

from the samples of the low-resolution image by means of some *a priori* model which incorporates information about features in the high-resolution image. The selected algorithm models the high-resolution image as a Gibbs-Markov random field and estimates the solution by means of a maximum *a posteriori* technique. It was found that this algorithm suffered from blurring. The cause of the blurring was isolated to being the general way in which the clique potentials are applied to every sample of the high-resolution estimate, regardless of local contours. This general approach allows smoothing across discontinuities. To eliminate the blurring, a set of directional weights were introduced that provide a control mechanism for preventing smoothing across discontinuities.

As a final note, this work was performed in parallel with the research of [38], in which a new adaptive interpolation algorithm is developed. This new algorithm is an iterative technique which estimates the high-resolution image using a total-variation method. The algorithm uses an objective function which minimizes the curvature of planar contours in the high-resolution estimate. Emphasis is placed on the correct modelling of the data fidelity term. A new technique for estimating the sensor model used in the acquisition of the low-resolution image is proposed.

1.3 Structure of the Thesis

Chapter 1 introduces the concept of adaptive image magnification and discusses the motivation for this work. A brief description of the selected algorithms and the proposed improvements are given.

Chapter 2 surveys the current research literature on the topic. Following a detailed elimination process, two algorithms are then selected for implementation.

Chapter 3 gives a brief overview of the selected edge-directed algorithm of [16] and discusses the motivation for the proposed improvements. Improvements to correct problems in the edge-detection and interpolation stages are then presented.

Chapter 4 gives a brief overview of the selected statistical algorithm of [29] and discusses

the motivation for the proposed improvements. Improvements to help eliminate blurring across discontinuities are then presented.

Chapter 5 presents a comparison of the selected and improved algorithms. The comparison provides several test images that were magnified with each respective algorithm, as well as the standard bicubic interpolation method.

Chapter 6 will provide a summary of the results of this research and outline possible avenues for future exploration on the topic.

Appendix A provides a detailed description of the theory and implementation of the selected edge-directed algorithm as described in [16].

Appendix B provides a detailed description of the theory and implementation of the improvements made to the edge-directed algorithm. The material in this appendix is based on the descriptions given in Section 3.2 of Chapter 3.

Appendix C provides a detailed description of the theory and implementation of the selected statistical algorithm as described in [29].

Appendix D provides a detailed description of the theory and implementation of the improvements made to the statistical algorithm. The material in this appendix is based on the descriptions given in Section 4.2 of Chapter 4.

Chapter 2

Literature Survey

The following provides a survey of the current literature on adaptive image magnification. To begin, a brief description of the information resources and the methodology used to conduct the search will be given. The results of the search will then be classified and the most promising among them will be presented. Finally, a choice of two algorithms designated for implementation will be made.

To simplify the review, the search results were sorted into five categories according to the approach taken by each respective solution. Papers within each category were reviewed and compared to each other. The final choice was made by comparing the best papers from each category and selecting the most promising among them from across the categories. The five categories are:

Linear: Techniques involving up-sampling followed by a linear filtering operation (i.e. bilinear, bi-cubic and spline interpolation).

Non-Linear: Techniques involving a non-linear optimization operation based on low-resolution image features (i.e. optimization of a PDE description of image intensity profile).

Transform: Techniques involving interpolation/extrapolation performed in the domain of a selected transform (i.e. Fourier transform, wavelet transform, Laplacian pyramid).

Edge-Directed: Techniques involving the use of edge information to guide the interpolation process (i.e. bilinear interpolation guided by an estimation of the edge mapping of the high-resolution image).

Statistical: Techniques involving the estimation of the high resolution image by the application of a stochastic model (i.e. Gibbs random field model applied with priors based on edges).

From these five categories, only the first is not an adaptive technique. Of the four remaining categories, the volume of papers indicates a research trend towards transform and edge-directed techniques. Non-linear and statistical techniques seem to have found greater use in highly specialized applications such as super-resolution, where a sequence of video frames are combined to form a single, high-resolution image, and medical imaging, where the algorithms are tied to the underlying physics of the image-acquisition process or are highly constrained by prior knowledge of image features. Although less common, there are interesting papers taking a non-linear or statistical approach to the more general area of image magnification.

2.1 Survey of Image Magnification Algorithms

2.1.1 Linear Methods

Linear techniques in the literature, [1]–[4], use linear space-invariant filters to interpolate the high-resolution samples. Common choices of interpolation filter are nearest neighbor, bilinear, bicubic, quadratic, Gaussian and various types of spline functions [1]. Since the theory behind linear interpolation is well established, most of the research on this approach is focused on finding new filters which reduce artifacts introduced by the traditional filters, as well as more efficient implementations. In [2], a modified version of the B-spline is used to obtain interpolation filters with better frequency responses, [3] proposes an FIR filter design method that attempts to account for the properties of human vision, and [4]

develops non-separable cubic convolution kernels to replace the traditional separable cubic filter. Due to the relative simplicity and efficiency of linear interpolation techniques, they are the most common approach provided by commercial software packages such as Adobe PhotoShop and Matlab.

2.1.2 Non-Linear Methods

Non-linear techniques in the literature, [5]–[7], use non-linear optimization processes constrained by certain image features. In [5], a method which optimizes a convex cost function based on an approximation of the gradient of the high-resolution image from the low-resolution image is presented. This method attempts to preserve edges by adding constraints on their orientation. A different approach is taken by [6], in which the problem is viewed from a geometric perspective. In this method, an image is first linearly interpolated. Then spatial regions of constant intensity are warped such that level curves are smoothed, thereby sharpening boundaries between regions. In [7], a regularized image interpolation method is proposed which focuses on the correct modelling of the image acquisition and display processes.

2.1.3 Transform Methods

Transform techniques in the literature, [8]–[15], are primarily focused on the use of multi-resolution decomposition, followed by interpolation applied to each level of the decomposition and/or extrapolation of higher resolution levels. These approaches aim at synthesizing the high frequency components of the magnified image by adapting the interpolation to suit the frequency content contained at each level of decomposition. In [8], higher resolution levels of a Laplacian pyramid decomposition are extrapolated from lower ones. Another approach, taken by [9], makes use of a filter bank which extracts edge directional components from the low resolution image and interpolates each sub-band in a directional specific way as to enhance the edges it contains.

2.1.4 Edge-Directed Methods

Edge-directed techniques in the literature, [16]–[28], use the edge information from the low-resolution image to obtain an estimate of the edge mapping of the high-resolution image. This estimate is then used to control some form of interpolation such that edges are not smoothed. The solutions differ mainly in the accuracy of the estimation of the edge mapping ([16] uses zero crossings resulting from a Laplacian operator to estimate the edge locations, whereas [17] uses a simpler gradient operator and [18] uses a local covariance measure) and the interpolation scheme ([16] uses variable order polynomials fitted according to edge content within a neighborhood, [17] chooses between three different types of polynomials depending on the location of a pixel with respect to edges and [18] uses local covariance to compute the coefficients in a weighted average of four low-resolution samples).

2.1.5 Statistical Methods

Statistical techniques in the literature, [29]–[31], attempt to estimate the high-resolution image based on the properties of the given low-resolution image. In [29], the high-resolution image is modelled by a Gibbs-Markov random field with specially selected clique potentials to classify the properties of each neighborhood. The chosen potentials allow the classification of pixels by degrees of smoothness or discontinuity, thereby being able to properly handle edges. Another approach creates a set of pixel classifications gathered from the statistics of pixels in typical training images [30]. Once trained, the algorithm interpolates an image by estimating the best filter coefficients (in the mean-square sense) for each neighborhood.

2.2 Selection of the Algorithms

Following the detailed review, the algorithms proposed by [16] and [29] were selected for implementation.

2.2.1 Eliminations

In the following, references will be eliminated in order to help clarify the selection of the two previously mentioned papers.

In [5], the variational description of the interpolation problem is well formulated. Unfortunately, the authors greatly simplified their solution with the goal of achieving an efficient interpolation by a factor of 2. Although the approach of [6] is interesting, the final result relies entirely upon the initial magnification, which is performed using a linear technique. This approach may serve best in the role of a post-processor used to remove staircase artifacts and enforce smoothness in regions of constant intensity.

The technique proposed in [9] uses a directional filter bank to isolate edges oriented along eight directions. This approach may encounter difficulties in large textured areas due to the non-uniform treatment of the texture caused by splitting the area into different sub-bands and interpolating each one differently. In [10], a linear MMSE estimator is used for synthesizing the finer detailed coefficients at the next level in a wavelet decomposition. [11] attempts to predict the LH, HL and HH sub-bands of the wavelet decomposition of the high-resolution image using a trained hidden Markov tree. [12] uses the energy in the wavelet coefficients located at an edge to extrapolate the finer detailed scales. Unfortunately, these three papers do not provide enough details allowing for an accurate reproduction of their results. [13] proposes an approach to interpolation that separates a particular set of features (objects, edges, textures, etc.) in an image, using a specialized filter bank, and interpolates each one in a feature dependent fashion. The high-resolution image is obtained by synthesizing the various features. Although the idea is very interesting, the paper assumes that the features have already been selected. It does not provide details regarding the implementation of such a system with regards to specific features.

The edge-directed algorithms proposed by [19, 20, 21] are superseded by [16]. They differ primarily in the methods used for edge detection and interpolation. An interesting approach developed in [22, 23] uses irregular sampling of a fitted surface to maintain edge sharpness. At an edge, this idea allows samples to be taken near the extremities of the

surface, which avoids interpolation across the edge. Unfortunately, since they both use linear interpolation methods (nearest neighbor, bilinear or bicubic), the irregular samples are still limited to a smooth, slow varying surface. Although this does allow for better separation at edges, the slope of the surface will still cause blurring. Other methods [24, 25, 26, 27] use edge directions to adapt the interpolation process. The main downfall of this approach is that the number of directions are limited (each paper uses only four directions), reducing the effectiveness of the algorithm for edges along general orientations.

2.2.2 First Selection

Compared to other edge-directed techniques, it was found that the algorithm proposed by [16] was both more complete in its description and more general in its application. The primary advantage of the proposed algorithm is that it interpolates by fitting polynomial surfaces of arbitrary order to account for discontinuities. This is in contrast to [17], which is limited to a choice of three polynomials, [18], which is an adaptive averaging filter, [20], which only modifies linear space-invariant interpolation techniques to become space-variant by adapting the filter coefficients to edges and [21], which post-processes a linearly interpolated image to enhance edges using a weighted average of neighboring pixels chosen using the Canny edge detector.

The choice of [16] was primarily based on experience with an implementation of the edge-directed interpolation algorithm proposed by [19]. In [19], an iterative edge-directed scheme uses an estimate of the edge mapping of the high-resolution image to guide a bilinear interpolation such that interpolation across edges is avoided. It was found that [16] dealt with all of the main flaws of [19]. To begin, [16] treats inaccuracies in the sub-pixel edge estimation process by using a rotationally invariant filter, eliminating problems with detecting edges of different orientations. Another improvement lies in using the edge zero-crossings in an analytical fashion, as opposed to quantizing the linearly interpolated edge estimates. This should help reduce the staircase edges and further errors in the estimation of edge locations. With regards to the rendering stage, [16] has improved upon

the interpolation of pixels near edges by fitting a variable order polynomial surface to the existing low-resolution pixels surrounding the edge. The order of the polynomial is determined by the magnitude of the discontinuity at the edge. This approach eliminates the use of the heuristic replacement procedure used by [19] which introduces errors by means of an inaccurate linear extrapolation of low-resolution pixels.

2.2.3 Second Selection

Compared to other statistical techniques, it was found that the algorithm proposed by [29] appears to be more robust and general in its modelling of image features. Although the ideas presented in [30] are very promising, the algorithm is dependent on the images used in the training sequence. This means that to properly train the interpolator, it is necessary to find training images that are representative of the type of images that will be commonly processed. Also, the training images must come in pairs of low- and high-resolution images. This can become problematic in situations where accurate high-resolution images are inaccessible. The approach taken by [31] is limited by constraints that enforce edges to lie along only four directions. Four directions are clearly not sufficient for an accurate description of edges in natural images or in the ability to properly handle textures.

The choice of [29] was based on the results presented by the authors of the paper. The authors formulated the problem of image magnification in the context of regularization, allowing them to draw on statistical estimation tools which incorporate *a priori* information into the model. To this end, they used a Gibbs-Markov random field to model the properties of pixels and their local neighborhoods. The authors chose the Huber function for the clique potentials, which allows the estimation problem to be reduced to the optimization of a convex objective function. This choice of clique potential adds constraints that enforce smoothness in regions of small intensity variations while not smoothing the discontinuities at edges. A compelling argument for this approach is in the use of the high correlation between neighboring pixels in an image. This allows introducing properties of

the physical process of image acquisition into the interpolation scheme. Also of particular interest is the incorporation of a Gaussian noise model into the estimation process. This integrates a certain degree of robustness into the magnification of degraded images. Further weight was given to this paper based on the success of Bayesian regularization techniques applied in the medical imaging field.

Also, it is interesting to note that the work done for [29] served as a precursor to work involving the extraction of a single high-resolution frame from a video sequence, with applications to scan conversion of interlaced video, done in [32]. In effect, [32] uses the same MAP estimation model as developed in [29]. It differs by the incorporation of motion estimation into the sub-sampling matrix and by the introduction of mixing parameters used to weight the influence of each frame in a sequence on the final high-resolution frame. The authors have incorporated the estimation of these new parameters into the same optimization algorithm used in [29]. Therefore, an implementation of [29] can be extended to the multi-frame case to obtain the algorithm from [32]. It is also interesting to note that the algorithm proposed in [32] was embedded within the MPEG video compression standard (as explained in [33]).

Chapter 3

Edge-Directed Algorithm

This chapter begins by providing a brief overview of the operation of the selected edge-directed algorithm, as described by the authors of [16]. Following the description, certain improvements to the algorithm will be proposed.

For a detailed description of the implementations, please refer to Appendix A, for the edge-directed algorithm, and Appendix B, for the improved edge-directed algorithm.

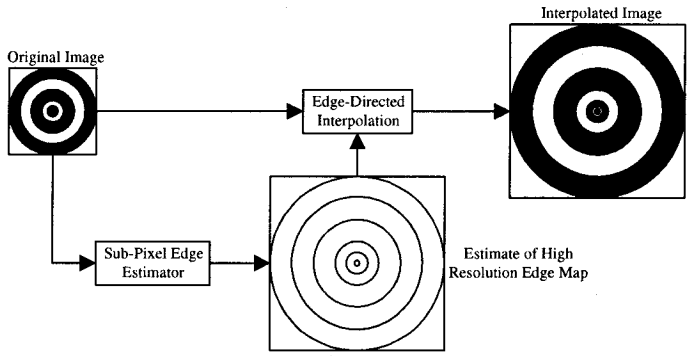
3.1 Algorithm Description

The idea of edge-directed interpolation is illustrated in Figure 3.1. Essentially, an estimation of the edge mapping of the HR image is used to guide an interpolation process such that interpolation across edges is avoided.

As determined during the literature survey, the most promising edge-directed algorithm is [16]. This algorithm can be divided into four stages: i) sub-pixel edge estimation, ii) preprocessing, iii) rendering, and iv) correction.

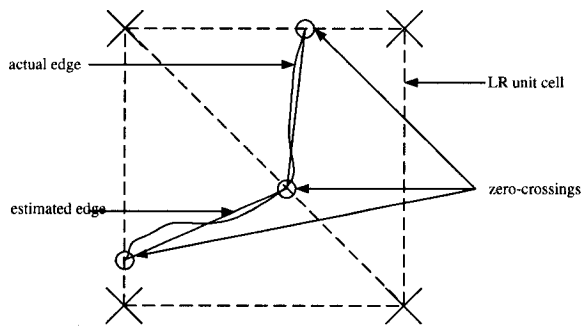
The sub-pixel edge estimation stage estimates the edge mapping of the HR image by linearly reconstructing edges based on the edge mapping of the LR image. The sub-pixel edge estimation stage has two components: i) the LR edge detection filter, and ii) edge estimation.

Fig. 3.1 Top-Level Block Diagram of the Edge-Directed Interpolation Scheme



The edge detection filter is a Center-On-Surround-Off (COSO) approximation to the Laplacian-of-Gaussian filter. This filter is designed to be rotationally invariant and have a DC response of zero. The filter is specified as having dimensions of 5x5. Refer to Section A.1.1 of Appendix A for the details involved in computing the filter coefficients.

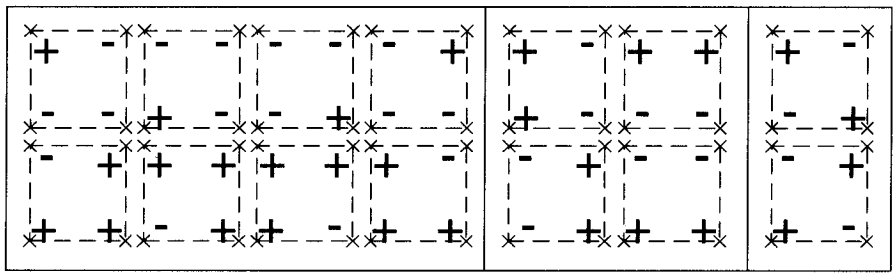
Fig. 3.2 Piece-Wise Linear Estimation of Edges Within an LR Unit Cell



Edge estimation attempts to reconstruct a piece-wise linear version of the actual edge, as shown in Figure 3.2. The concept relies on using the filtered LR image to find zero-crossings that indicate the presence of an edge passing through the LR unit cell. These zero-crossings are then linked by straight line segments to obtain an estimate of the edge.

The zero-crossings are determined by first applying the COSO filter to the LR image. Then, for every LR unit cell in the filtered image, the signs of the four corners are compared

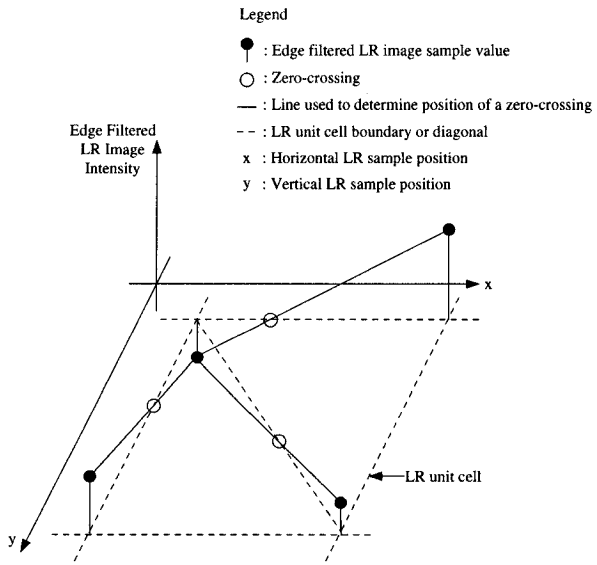
Fig. 3.3 Sign Configurations: a) Corner edges with 3 zero-crossings, b) Vertical and horizontal edges with 2 zero-crossings, c) Intersecting edges with 4 zero-crossings



to each other to find differences. A zero-crossing exists between two filtered samples if they have different signs. There are three basic sign configurations (shown in Figure 3.3): i) one corner has a different sign from the other three (part a of Figure 3.3), ii) two adjacent corners have a different sign from the other two corners (part b of Figure 3.3), and iii) two diagonal corners have a different sign from the other two (part c of Figure 3.3). In the first case (part a), an edge separates one of the four corners from the other three. In the second case (part b), an edge separates the unit cell in a horizontal or vertical fashion. In the third case (part c), all four corners are separated from each other by two intersecting edges. Once the sign configuration has been determined, the coordinates of the zero-crossings are estimated by finding the position where the lines joining two corners of different signs is zero. An example is depicted in Figure 3.4, for the leftmost sign configuration in part a of Figure 3.3. Once the zero-crossings have been located, the edge is estimated by linking two adjacent crossings by a straight line, resulting in a piece-wise linear edge. Refer to Section A.1.2 of Appendix A for further details on estimating zero-crossings.

It is important to note that a global threshold is not applied to the filtered LR image prior to the estimation of the zero-crossings, as is common practice with second-order operators. The purpose of the threshold is to eliminate edges resulting from noise or weak edges present in the background features of an image. Conventional edge detection is oriented towards finding contours which delimit objects in an image. In this case, only

Fig. 3.4 Estimating Zero-Crossings from the Second-Order Edge Filtered LR Image



foreground edges are important. Weak edges in background objects tend to clutter the edge mapping.

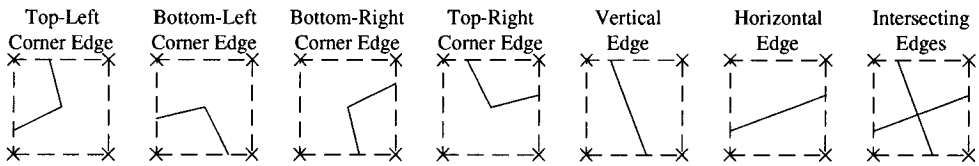
In contrast, image magnification is concerned with obtaining the most accurate and detailed edge map possible in order to give the interpolation scheme the best chances of producing quality results. In this case, all edges are considered important. Although some edges may result from noise, weak edges will not be eliminated. If noise in the LR image is a concern, a de-noising technique should be applied prior to magnification.

The preprocessing stage is used to correct possible errors in the estimation of the HR edge mapping. The piece-wise linear estimation of edges creates false separations between HR samples within LR unit cells. This is due to the fact that the zero-crossings are only computed using the four corners of the filtered LR cell. This results in a small number of zero-crossings, which are insufficient to obtain a good estimation for the actual edge. The approach taken by [16] computes a replacement value for samples of the filtered LR image such that the zero-crossings are moved. Only samples that are part of LR cells containing edges are replaced. The replacement value for a sample in the filtered image is computed

as the mean of the eight LR nearest neighbors. If the mean is zero, then the median is used.

The rendering stage uses the estimated HR edge mapping to guide the interpolation. The new HR samples inserted into each LR unit cell are interpolated based on their position with respect to the edge passing through the cell, as well as the type of edge.

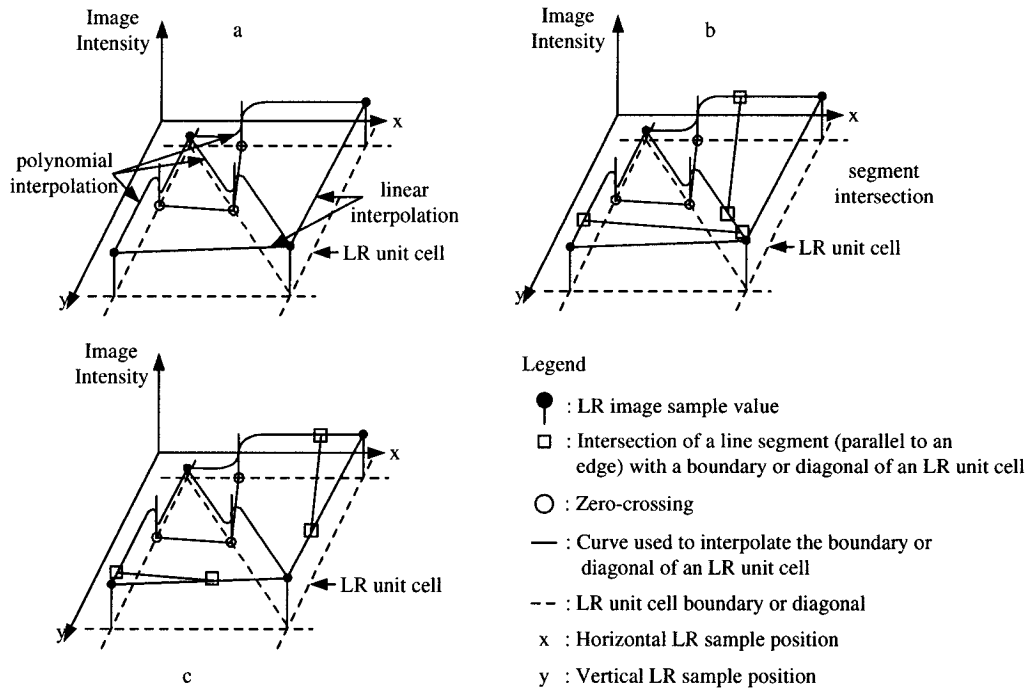
Fig. 3.5 Types of Edges Passing through an LR Unit Cell



The interpolation is performed using each group of four LR samples. If there is no edge present within an LR unit cell, bilinear interpolation is applied. If an edge is present, a variable order polynomial surface is fitted to the cell such that it interpolates the four LR corner samples of the cell. The order of the polynomial is determined by a heuristic which makes use of the magnification factor and the magnitude of the discontinuity at the edge between samples. The polynomial surface is fitted differently for each type of edge. From the sign configurations used during edge detection, there are seven distinct types of edges (shown in Figure 3.5).

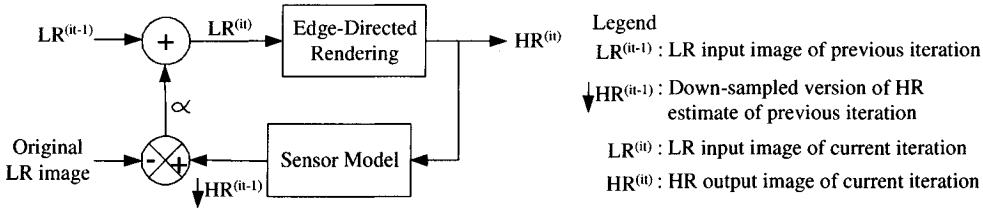
For each edge type, a variable order polynomial curve is fitted between any two corners of an LR unit cell separated by a zero-crossing. Corners not separated by a zero-crossing are interpolated linearly. The HR samples contained within the LR unit cell are interpolated linearly using the boundaries or the diagonals of the LR unit cell as end points. For every sample in the interior of the cell, a line segment parallel to the edge is determined. This line segment intersects in at least two locations with the boundaries or diagonals of the cell. The two intersections nearest the sample being interpolated are used as end points for the linear interpolation. The value of the end points are determined by the curve fitted to the boundaries or diagonals. Clearly, the geometry of each edge type must

Fig. 3.6 Edge Interpolation: a) Interpolating the HR samples on the boundaries of an LR unit cell, b) Interpolating the HR samples within an LR unit cell (case 1), c) Interpolating the HR samples within an LR unit cell (case 2)



be considered separately. Figure 3.6 illustrates the process step-by-step in the case of a corner edge (leftmost sign configuration of part a of Figure 3.3). Part a of Figure 3.6 shows how the variable order polynomial edge curve is fitted to the top, left and diagonal boundaries of the LR unit cell where zero-crossings are present. Since the bottom and right boundaries do not have any zero-crossings, linear segments have been fitted. Parts b and c of Figure 3.6 illustrate the strategy for interpolating HR samples in the interior of the LR unit cell. In both cases, a line segment parallel to the edge is used to linearly interpolate HR samples in the interior of the LR unit cell. The two cases differ in the location of the intersections with the boundaries. A distinction has been made in order to clarify the steps needed to compute the values at the intersecting end points of the line segment parallel to the edge. Refer to Section A.1.4 of Appendix A for further details on rendering.

Fig. 3.7 Block Diagram of the Edge-Directed Correction Stage Used to Successively Approximate the HR Estimate



The correction stage attempts to repair any damage to the original image data that may have been incurred during the edge-directed interpolation process. The term damage is meant to signify anything resulting from the interpolation which causes the HR estimate to severely deviate from the actual HR version of the LR image. Naturally, since the actual HR version is unknown, this stage can only perform corrections according to an assumed model representing the desired properties of the actual HR version. In this case, the model is assumed to be the standard moving average filter used to simulate the image acquisition process of modern digital sensors. The damage is corrected by using a feed-back loop which successively approximates the HR estimate. The goal of the feed-back loop is to minimize the error between the original LR image and a decimated version of the HR estimate. The idea is that when the error reaches zero (i.e. the decimated HR estimate matches the original LR image exactly), the HR estimate has been reconstructed to match the actual HR version as closely as possible (given the assumed sensor model). Although the HR estimate may not be unique, it is assured to be a valid representation of the actual HR version. The degree to which the LR image input to the rendering stage is corrected is determined by the parameter α. This parameter is a weight which defines the step size of the applied change at each iteration. The correction stage is illustrated in Figure 3.7. At each iteration *it*, the LR input is obtained by correcting the LR input from the previous iteration (*it-1*) with the weighted difference between the down-sampled version of the HR estimate from the previous iteration and the original LR image. Convergence of the algorithm was proven in [37]. Refer to Section A.1.5 of Appendix A for further details

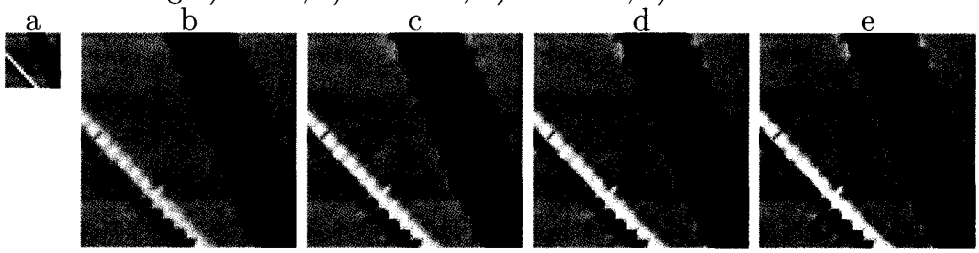
on correction.

3.1.1 Algorithm Parameters

The choice of this algorithm's single parameter, α, controls the degree to which the LR image being magnified is corrected. This regularization parameter is the gain applied to the feedback loop used to perform the successive approximations. Note, that although not an explicit parameter, the maximum number of iterations can also be used to control convergence.

The images in Figure 3.8 illustrate the effect of α. The same test image was used in order to provide a better comparison of the effects of varying the parameter.

Fig. 3.8 Effects of Varying the Regularization Parameter of the Edge-Directed Algorithm (16× Surface Magnification): a) Original LR image, HR estimate using b) $\alpha = 0$, c) $\alpha = 0.25$, d) $\alpha = 0.75$, e) $\alpha = 1$



The original LR image is shown in part a) of Figure 3.8. Note that all HR estimates were performed for 10 iterations.

Images b) to e) show the effect of allowing α to vary. Observe how increasing α introduces significant ringing around edges. The ringing is caused by the use of the moving average sensor model.

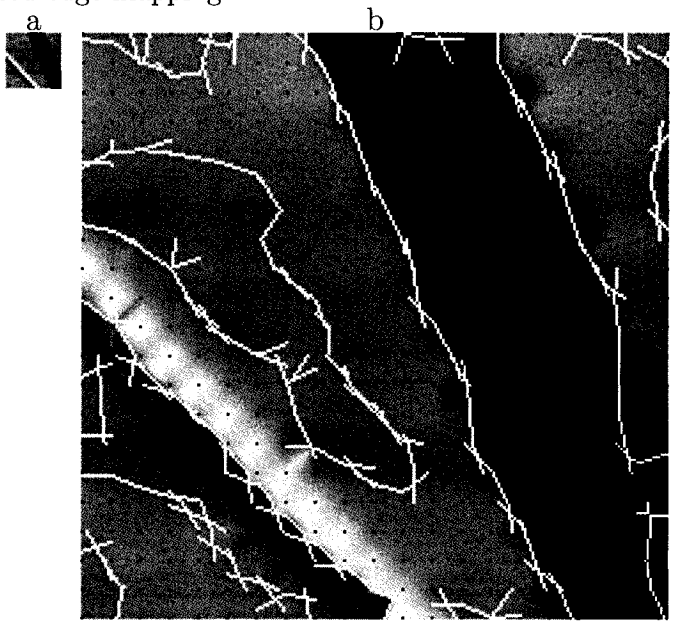
3.1.2 Algorithm Observations

Results from the implementation of the edge-directed algorithm of [16] revealed that the algorithm has difficulty in correctly estimating edges to sub-pixel accuracy. This is shown in Figure 3.9, where the original LR image is shown on the left, and on the right, is the

edge-directed HR estimate for a factor of 100× surface magnification. The estimated HR edge mapping has been overlayed onto the HR estimate in order to clearly show the edge information being used during interpolation.

This particular LR image was selected due to its edge content, which boasts narrow edges of one sample in width (in the bottom-left area), as well as wide edges of more than one sample in width (in the top-right area). The regularly spaced black samples indicate the positions of the original LR samples and the white lines are the estimated edges.

Fig. 3.9 Difficulties with the Edge-Directed Algorithm (100× Surface Magnification): a) Original LR image, b) Edge-directed HR estimate overlayed with estimated edge mapping



There are three important observations to be made from Figure 3.9. The first observation is that the estimated edges do not coincide with the narrow edges. In fact, the estimated edges are shifted by one LR unit cell away from the correct location. Clearly, the edge detection filter does not possess sufficient sensitivity for accurate edge localization.

The second observation is that many LR unit cells are improperly classified. From Figure 3.3, a unit cell with an edge must have two or four zero-crossings on its borders. However, in Figure 3.9, there are many cells with a single zero-crossing. This occurs when

an adjacent unit cell is classified as having four zero-crossings instead of two, leaving a single zero-crossing, that should not be present, along a cell boundary.

A third observation can be derived from the first two. If the location of the edges are not properly known, the rendering will be faulty. This is due to the fact that the rendering is blind to the actual features of the image. It is only responsible for the interpolation of the samples on the HR grid. It can not perform any manner of correction. Adding complexity to the rendering stage will not help in correcting errors in the estimated edge map. Therefore, a simpler strategy should be used for the rendering stage.

3.2 Improved Algorithm

This section proposes methods for improving the sub-pixel edge estimation and rendering stages of the edge-directed algorithm (as discussed in Section 3.1.2).

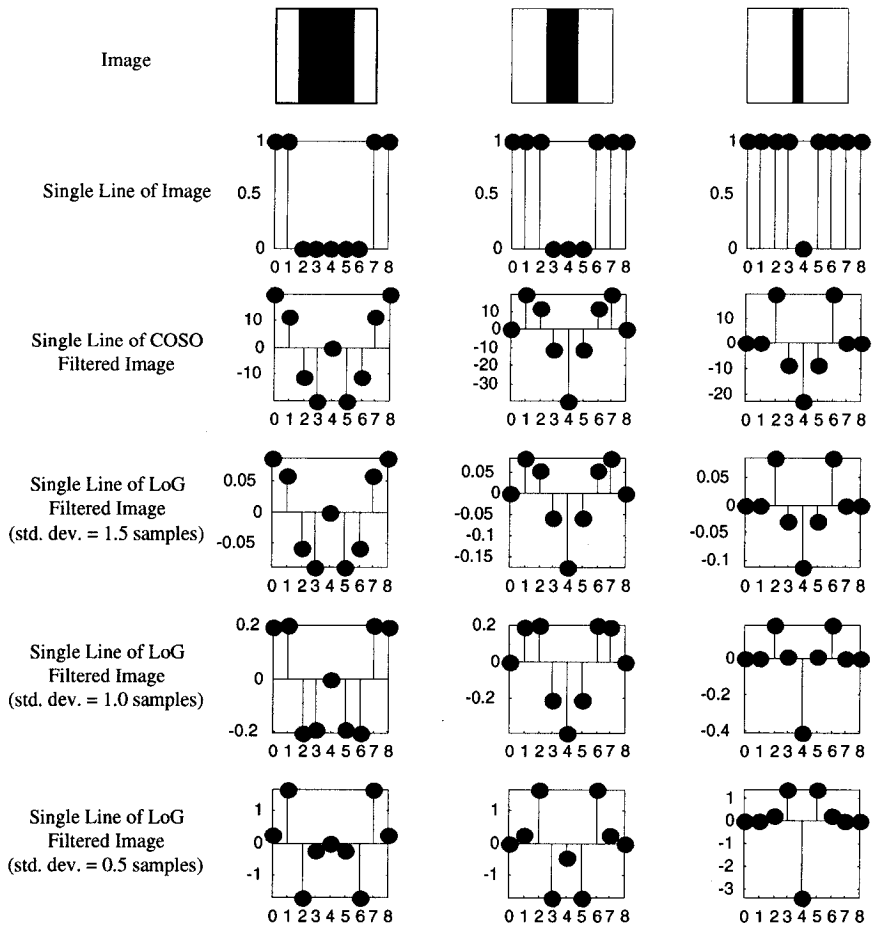
3.2.1 Improved Sub-Pixel Edge Estimation

Zero-Crossing Estimation

The first step in improving the accuracy of the edge estimation is to take a closer look at the filter used for detecting the presence of edges in the LR image. The filter designed in [16] is a circularly symmetric version of the COSO filter proposed by [19]. This filter, specified by a constant, positive disk surrounded by a constant, negative, ring, is essentially a thresholded version of the standard Laplacian-of-Gaussian (LoG) filter. Following experimentation, it was found that the COSO filter was unable to properly track edges that were too narrow. To demonstrate, Figure 3.10 illustrates the result of filtering a set of ideal step edges using the COSO and LoG filters (each filter has dimensions 5×5).

For the first two step edges, both filter types are able to properly estimate the location of the transitions (indicated by sign changes between samples in the filtered image). However, in the case of a step edge of one sample wide, only the LoG filters of standard deviation 1.0 and 0.5 are able to correctly estimate the transitions. It can be seen that the

Fig. 3.10 Accuracy of the Edge Detection Filter for Ideal Step Edges



COSO filter actually shifts the location of the edge by one sample away from the correct location, which widens the one sample step edge to three samples. Clearly, the low-pass filtering being performed by the COSO filter is excessive. Furthermore, the test images represent an ideal artificial case. For natural images with varying gray levels and noise, the performance deteriorates significantly by causing shifts in edge locations of two or three samples. In many cases, edges are smoothed to the point of non-existence.

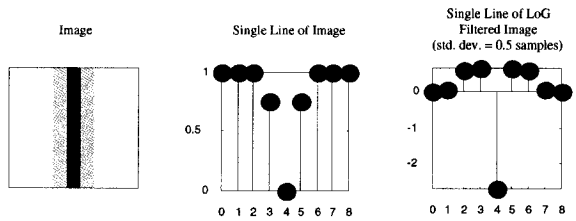
From Figure 3.10, it can be observed that the result of applying the COSO and LoG of standard deviation 1.5, are very similar. In fact, they differ primarily by a scaling factor.

To achieve better accuracy in the localization of edges, the LoG filter of standard

deviation 0.5 will be used. From Figure 3.10, the LoG filter of standard deviation 1.0 is just barely able to detect the one sample wide step edge. Therefore, the LoG of standard deviation 0.5 would have the best chance of properly estimating the location of edges in natural images. Also, it is important to note that noise immunity in image magnification is less important than the accurate localization of edges. Although a sensitive edge detection filter may consider noise to be an edge, it is of greater importance to ensure that edges are not missed. Even false edges, caused by nearly homogenous regions located close to a transition (as in the first step edge of Figure 3.10) are tolerable. Although the edge may be false, the edge-directed rendering will model the edge with a straight line of slope zero, reducing the interpolation to the case of no edge.

The dimensions of the LoG filter were experimentally chosen to be 5×5. It was found that a smaller filter produced errors in the position of zero-crossings due to the lack of local information resulting from it's small coverage area. Larger filters tended to miss edges, as well as shift them, due to over-smoothing caused by incorporating too much information beyond the local edge.

Fig. 3.11 Inaccurate Detection of Weak Edges in the Presence of Stronger Edges



Although the LoG filter has increased the accuracy of the edge localization, there are certain situations in which it fails to properly detect edges. These situations arise when a weaker edge is in the presence of a stronger one. For example, consider Figure 3.11, in which a one sample wide shadow has been added to the step edge. This shadow should be interpolated as an edge and would be if the stronger edge were not present. However, the strong edge has the effect of biasing the edge detection in its favor. Unfortunately, a

solution to this problem was not found.

Edge Classification

Following the filtering, the position of edges are found by estimating the zero-crossings between all pairs of samples of different signs. Then, the edge passing through an LR unit cell is classified according to its type and topology. The type of an edge defines how the LR corner samples are separated by the edge. To determine the edge type, it is only necessary to compute the four zero-crossings along the borders of an LR unit cell. From the possible sign configurations resulting from the filtering operation (Figure 3.3), only configurations with two or four zero-crossings on the borders are possible. The topology specifies how the zero-crossings are linked to form the edges. Figure 3.5 illustrates the edge types used by [16]. In the case of the four corner edge types, the topology is determined by the diagonal zero-crossing. The other edge types only admit one topology.

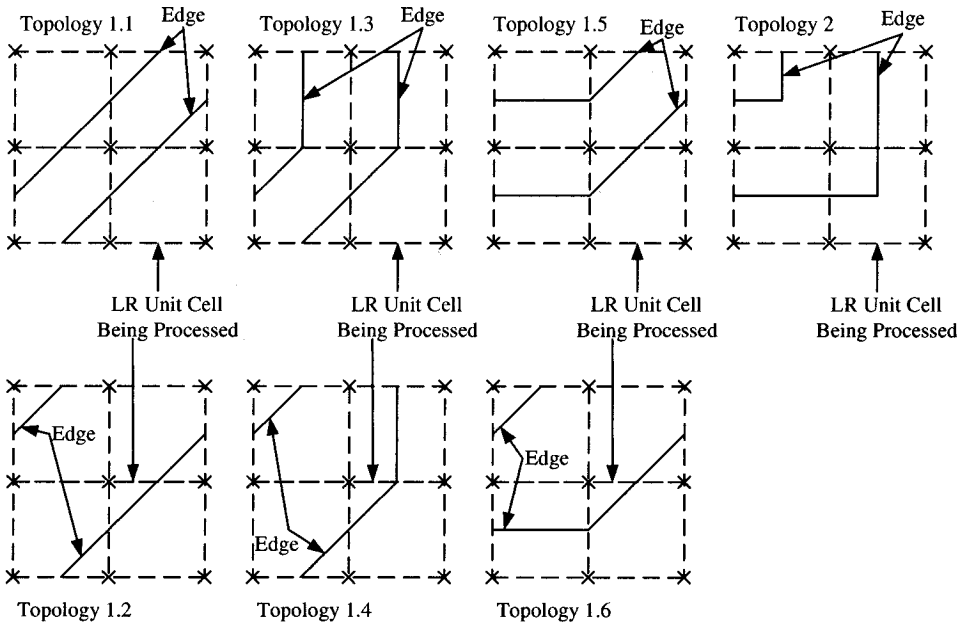
It was found that zero-crossings in the horizontal and vertical directions were far more accurate than in the diagonal directions. Zero-crossings in the diagonal directions tend to overshoot the position that visual inspection would claim to be the correct one, causing edges to become jagged. To correct this, [16] uses a pre-processing stage which executes a replacement strategy on the filtered LR image. However, following implementation, it was found that the correction is not uniform. Although the location of certain diagonal zero-crossings are improved, others that were correct prior to the pre-processing become incorrect. Overall, no noticeable improvement was made.

Due to the difficulty in accurately estimating zero-crossings in the diagonal directions, it was decided to only use the zero-crossings in the horizontal and vertical directions. This change does not affect the determination of the edge type, however a new strategy for estimating the topology will have to be introduced.

In the following, the new topology estimation strategy will be outlined for the case of a corner edge type. This strategy will then be generalized for the case of four zero-crossings.

The new topology strategy must avoid introducing jagged edges in areas where the

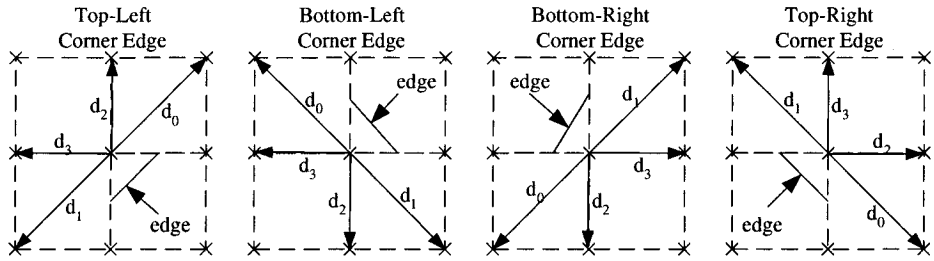
Fig. 3.12 Edge Topologies for a Top-Left Corner Edge Type



edge should be straight, while not straightening sharp corner edges. To accomplish this, the new strategy will need to estimate the direction of the edge by finding trends in the local area of the isolated corner. Figure 3.12 illustrates the possible topologies in the case of a top-left corner edge type.

By observation, the direction of an edge can be determined by classifying the local variations between the corner sample and some of its nearest neighbors. Figure 3.13 illustrates the quantities of interest. Every d_i is the difference between the corner sample and the local sample indicated by the arrow. The trends indicating the direction of an edge are given by the magnitude of each d_i. Small magnitudes indicate the similarity of the local samples to the corner sample, which in turn indicate the high probability that a local sample is on the same side of the edge as the corner sample. Large differences indicate that the corner and local samples are on opposite sides of the edge. Note that two of the eight nearest neighbors are not included in the set of d_i's. This decision was made in order to properly classify topologies for edges that are only one sample wide. For

Fig. 3.13 Quantities Used in Estimation of the Edge Topology



such edges, looking across the edge would not provide a valid estimate of the direction of the edge. Edges that are wider than one sample are not affected by this decision, since the topology taken from the perspective of the corner sample should not recognize edge width, only direction.

From the above, a measure of the similarity between the corner sample and a local sample can be expressed as

$$\delta_i = 1 - |d_i| \tag{3.1}$$

where $i = 0, 1, 2, 3$ and the d_i are as indicated in Figure 3.13.

Table 3.1 Edge Topology Matrix (**T**)

	δ_0	δ_1	δ_2	δ_3
Topology 1.1	1/2	1/2	-1/2	-1/2
Topology 1.2	1/4	1/4	1/4	1/4
Topology 1.3	-1/2	1/2	1/2	-1/2
Topology 1.4	-1	1/3	1/3	1/3
Topology 1.5	1/2	-1/2	-1/2	1/2
Topology 1.6	1/3	-1	1/3	1/3
Topology 2	-1/2	-1/2	1/2	1/2

To classify a particular configuration of the δ_i as a topology, a simple matrix can be used to perform correlation with respect to the ideal edge types in Figure 3.12. Let **T** be

the 7×4 matrix which classifies the local trends 4×1 column vector, $\delta = [\delta_0, \delta_1, \delta_2, \delta_3]^t$, into two distinct topologies: i) a straight line between the two zero-crossings, and ii) a sharp corner formed by the intersection of horizontal and vertical line segments. Table 3.1 provides the coefficients of the matrix for each of the possible topologies in Figure 3.12. Positive coefficients reward local samples that are on the same side of the edge as the corner sample, whereas negative coefficients penalize local samples that are on opposite sides of an edge. The same matrix can be used for the other three corner edge types by simply rotating the LR unit cell along with the definition of the d_i (see Figure 3.13).

The scaling coefficients in matrix $\mathbf{T}$ were chosen empirically such that measures along the direction of the edge are averaged together. Similarly, measures across an edge are also averaged. The goal of the averaging is to introduce robustness. It was found that using the raw measures directly had a tendency of biasing the results towards incorrect topologies when gray values along the direction of an edge did not exactly match. Since grey values will vary in natural images, it is important to take into consideration the fact that samples on an edge may themselves be changing. The importance lies in the relative magnitudes of these changes.

The topology is determined by the row with the maximum of the correlation, $\mathbf{T} \cdot \delta$. Note that Figure 3.12 contains seven topologies in total. However, viewed from within the LR unit cell, all of the topologies are of the straight line type except the last. Hence the labelling, topology $1.x$, where x represents a variant of the straight line topologies. The sharp corner topology (topology 2) has no variants. Therefore a maximum at any topology other than the sharp corner should be considered a straight line. Figure 3.14 shows the set of edge types and topologies for two zero-crossings. The procedure for determining the edge topology in the case of two zero-crossings can be summarized as follows:

1. Apply the 7×4 topology matrix, $\mathbf{T}$, to the 4×1 column vector δ of local trends. The product is $\mathbf{t}_\delta = \mathbf{T} \cdot \delta$.

2. Estimate the topology of the corner edge by finding the row of $\mathbf{t}_\delta$ with the maximum

correlation. All topologies are of the straight line type unless topology 2 is the maximum.

Fig. 3.14 Expanded Set of Edge Types and Topologies in the Case of Two Zero-Crossings

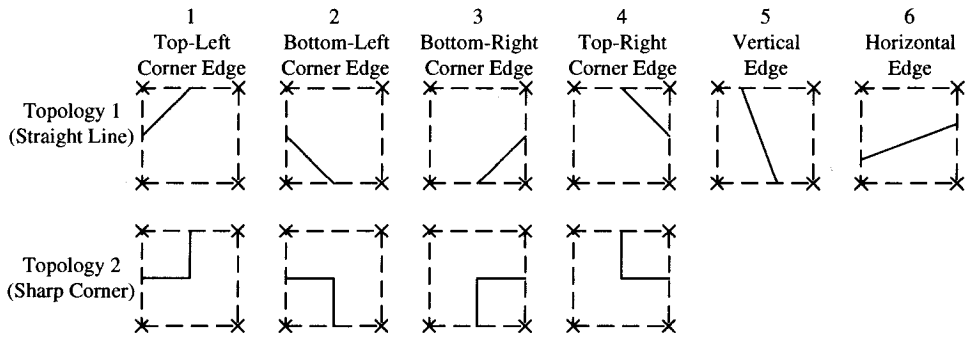
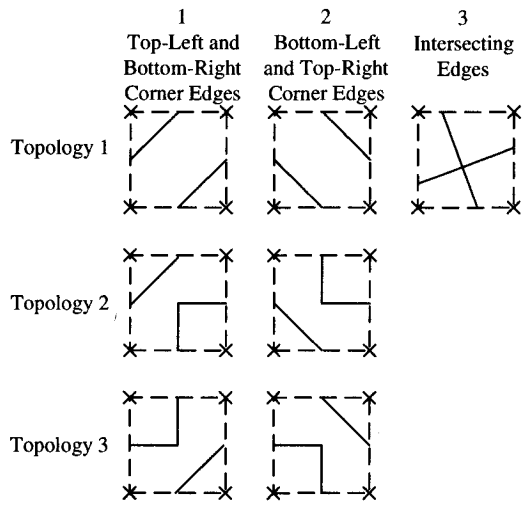


Fig. 3.15 Expanded Set of Edge Types and Topologies in the Case of Four Zero-Crossings



In the case of an LR unit cell with four zero-crossings, the above strategy will be applied to each of the four corners independently. This will result in an expansion of the set of edge types and topologies for four zero-crossings. Figure 3.15 illustrates the expanded set of edge types, along with the possible topologies for each type. Once the topologies have

been determined for each of the four corners, they must be combined to allow classifying the edge type.

The procedure is as follows:

1. Apply the 7×4 topology matrix, $\mathbf{T}$, to a 4×4 matrix $\mathbf{\Delta}$ whose j^{th} column is the local trends vector δ^j for the j^{th} corner. The corner index j is taken in the anti-clockwise direction starting at the top-left corner. The product is $\mathbf{T_\Delta} = \mathbf{T} \cdot \mathbf{\Delta}$.

2. Estimate the topology of each corner by finding the row with the maximum correlation.

3. Form a 1×4 row vector, $\mathbf{t_{\Delta max}}$, whose elements are the value of the maximum topology correlation at each corner (i.e. the maximum of the row elements in each column of $\mathbf{T_\Delta}$).

4. If the topologies at all four corners are sharp corners, then the edge type is classified as intersecting edges (edge type 3 in Figure 3.15). Otherwise, apply the 4×2 edge type classification matrix $\mathbf{E}$, given in Table 3.2, to the 1×4 row vector $\mathbf{t_{\Delta max}}$. The edge type is determined by finding the column with the maximum. The column numbers correspond directly to the edge types in Figure 3.15.

5. Once the edge type is known, the topology is determined by simply combining the individual topologies of the corners to form one of the possibilities from Figure 3.15.

Table 3.2 Edge Type Classification Matrix in the Case of Four Zero-Crossings ($\mathbf{E}$)

	Edge Type 1	Edge Type 2
Maximum of Topology Correlation for Top-Left Corner	1/2	-1/2
Maximum of Topology Correlation for Bottom-Left Corner	-1/2	1/2
Maximum of Topology Correlation for Bottom-Right Corner	1/2	-1/2
Maximum of Topology Correlation for Top-Right Corner	-1/2	1/2

As a final note, this new sub-pixel edge estimation scheme no longer requires the pre-processing stage used by [16] to correct errors in the estimation of diagonal zero-crossings.

3.2.2 Improved Edge-Directed Rendering

The approach to the edge-directed rendering stage taken by [16] is quite complex. This is due to the fact that each sample on the HR grid contained within an LR unit cell with an edge passing through it must be fitted to a line segment which is parallel to the edge and passes through the coordinates of the sample. Then, the intersection of this line with the nearest borders and/or diagonals of the cell must be computed. Finally, the sample is interpolated linearly between the values at the intersecting end-points. The final result is that each LR unit cell is fitted with a surface that maintains borders and/or diagonals with edges and everywhere else is interpolated smoothly based on which side of an edge the sample is located.

This same effect can be achieved using a modified version of the simpler approach developed in [19]. The authors of [19] apply bilinear interpolation to groups of samples, as opposed to treating each sample individually. The groups are collections of samples that reside in regions separated by the edges passing through an LR unit cell. For all samples in a region, bilinear interpolation is applied by determining replacement values for the LR corner samples that are not contained in the region. The replacement strategy uses linear extrapolation based on the values of LR samples from the surrounding unit cells.

The primary flaw in this approach is that the replacement strategy needs to look outside of an LR unit cell in order to find replacement values. This can cause problems when surrounding cells also have edges between the LR samples. In such cases, the replacement strategy becomes ambiguous.

To avoid the necessity of using adjacent cells for computing the replacement values, a scheme was developed which combines the variable order polynomial edge modelling of [16], with the edge separated region interpolation of [19]. The idea is to compute the replacement values for the LR corner samples by linearly extrapolating between the LR

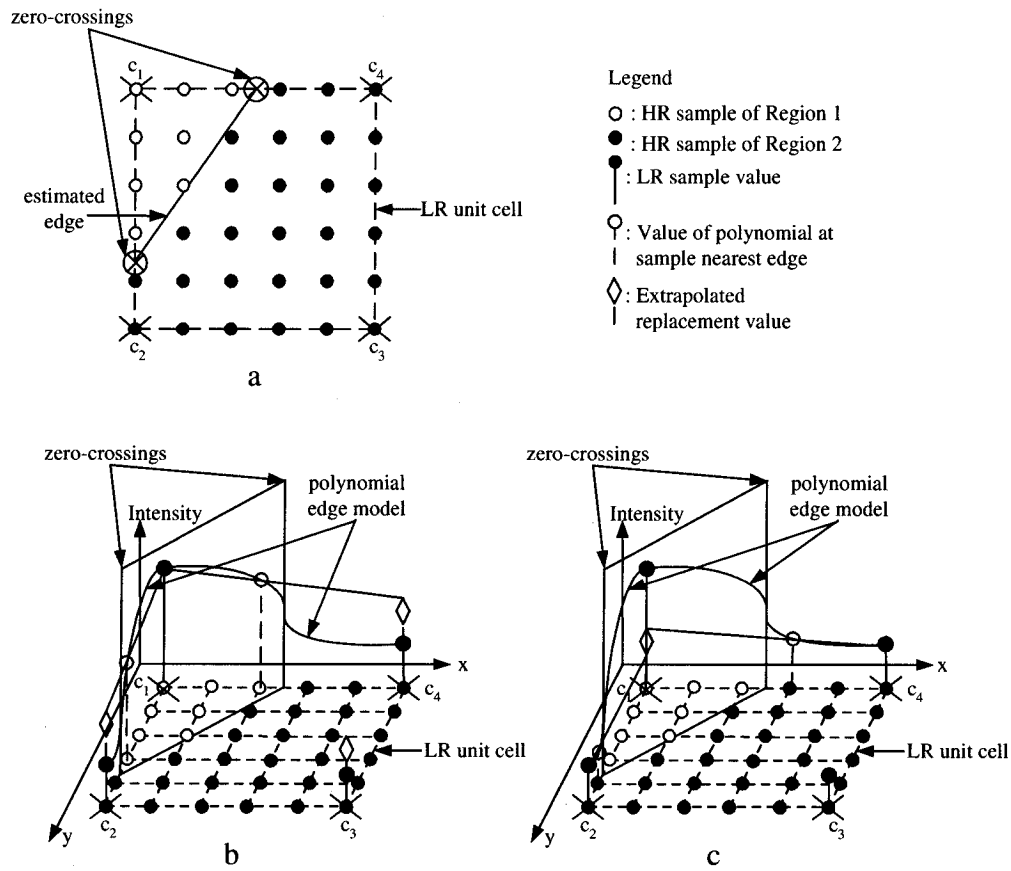
corner samples contained in the region and the value of the polynomial at the HR sample that is nearest the edge and also contained in the region. Since the location of the edge is given by the estimated zero-crossing, the HR sample nearest the edge and contained in the region can be found by simply rounding the zero-crossing towards the LR corner sample in the region.

Since the improved sub-pixel edge estimation described in Section 3.2.1 only estimates zero-crossings in the horizontal and vertical directions, edge types that have a single corner separated from the other three can only perform the linear extrapolation along those two directions. This leaves the replacement value for one corner undetermined. To determine this value, the replacement values at the two other corners can be averaged. This is a simple way of ensuring that smoothness is maintained along outer cell borders that do not have edges.

The interpolation of two regions is illustrated in Figure 3.16 for the case of a corner edge. Part a of Figure 3.16 gives a topological view of the LR unit cell and the edge passing through it. Note that the edge separates the cell into two regions. Part b of the figure illustrates the methodology used for computing the replacement values for the LR corner samples labelled c_2, c_3 and c_4. For example, the replacement value for c_4 is obtained by a linear extrapolation between c_1 and the HR sample nearest the zero-crossing at the top border of the unit cell. The replacement value for c_3 is obtained by averaging the replacement values for c_2 and c_4. Part c illustrates the same process for the second region. In this case, the replacement value for c_1 is the average of the linear extrapolations between c_4 and the HR sample nearest the zero-crossing at the top border, and c_3 and the HR sample nearest the zero-crossing at the left border.

The edge model used is a variation of the one developed by [16]. In [16], edges are modelled by a variable order polynomial composed of two segments joined at the zero-crossing. Each segment uses samples from surrounding cells to determine the slope at the corner points and relies on a heuristic to determine the order. The order of each segment is the same and is tied to the magnification factor.

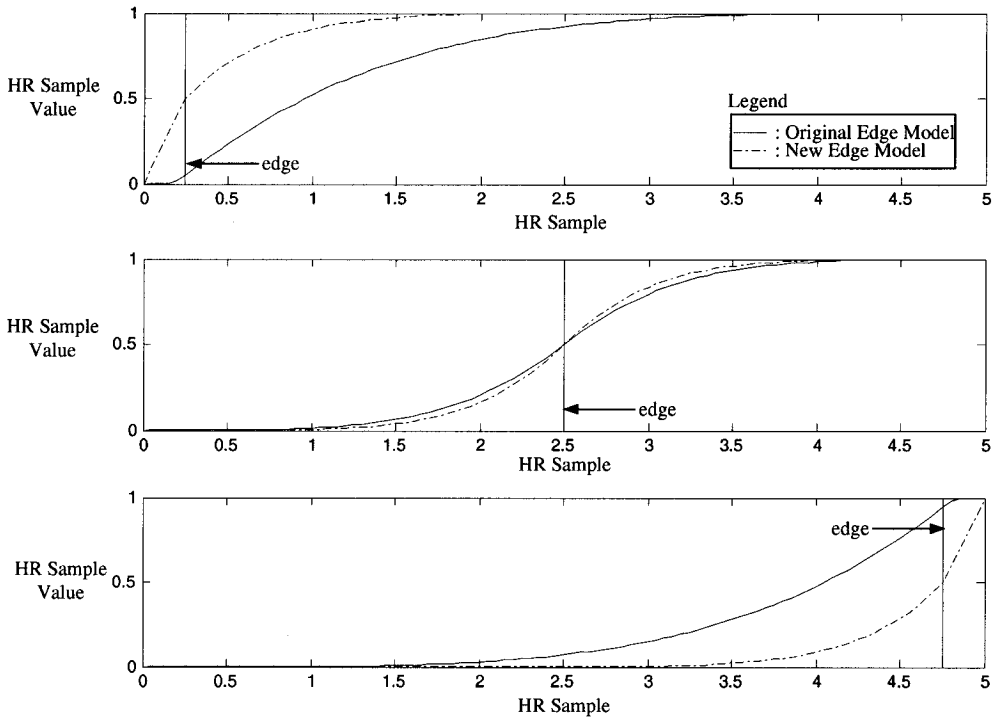
Fig. 3.16 Improved Edge-Directed Interpolation Scheme



It was found that the two curve segments were unable to properly bend when the zero-crossing approached one of the corners. This is due to the heuristic, which limits the maximum order of the polynomial segments to the magnification factor. By observation, as the zero-crossing approaches one of the corners, the order of the longest polynomial segment should be large in order to allow it to bend sharply, while the order of the shorter segment can be lower due to its more gradual transition. This provides motivation to correct the problem by allowing each curve segment to have its own order, allowing them to bend differently according to need. The order of each segment is based on the distance of the zero-crossing from each respective corner and the slope at the zero-crossing. It is

computed such that continuity of the slope at the zero-crossing is maintained. Figure 3.17 compares the bending ability of the two edge models as the zero-crossing approaches the corners. Note how the new edge model can bend more sharply. Also, this test was conducted with the maximum slope at the zero-crossing set to one. Larger slope settings will allow for even sharper bends.

Fig. 3.17 Comparison of the Original and New Variable Order Polynomial Curves Used to Model Edges

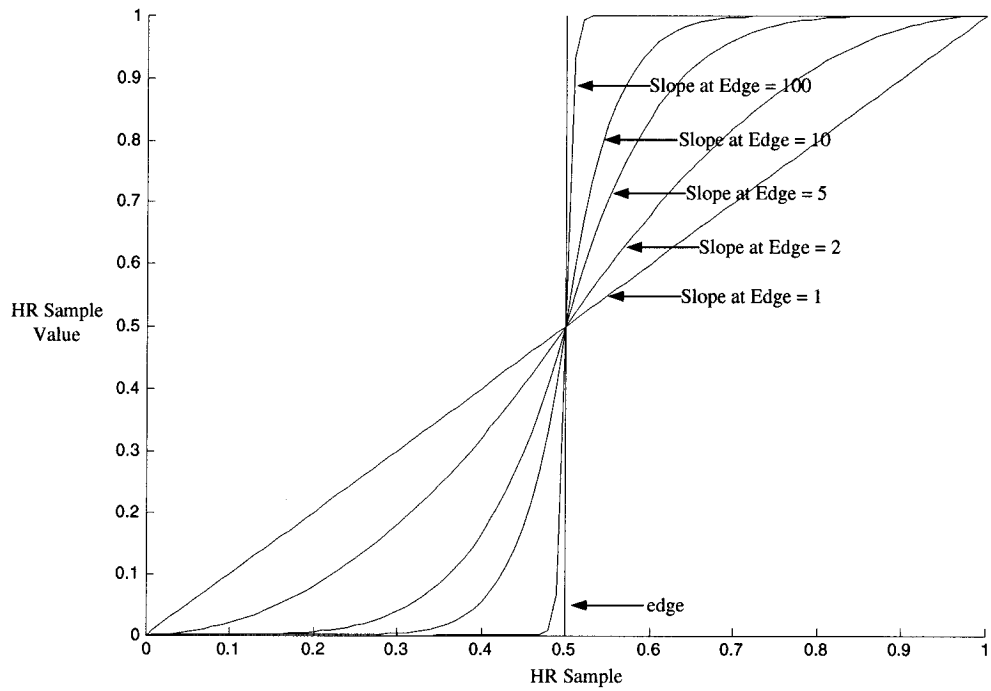


The slope at the zero-crossing is specified by applying a slope control function to the difference between the values at the two LR corner samples. The selected slope control function was determined by observation to have the following properties. When the step between the two corners is small, the slope should be almost linear. When the step is large, the slope should approach some specified maximum value. Following experimentation with linear, quadratic and cubic control functions, the cubic function was selected. It was found that the S shape of a cubic polynomial whose end-point derivatives are zero, provided the

best separation between regions of high contrast.

The maximum slope, labelled S, has been made a parameter of the algorithm in order to allow control over the sharpness at edges. When S is small, the polynomial curves approach straight lines, and when it is large, the curves tend towards ideal step functions. Therefore, the larger S, the sharper the edges. Specifying a large slope is equivalent to performing an edge-directed version of the zero-order hold interpolation.

Fig. 3.18 Demonstration of the Bending Ability of the New Edge Model



In order to avoid looking outside the cell to compute the slopes at the corners, it was decided to set them to zero. This decision removes complications in dealing with how to compute slopes when edges are present in adjacent cells. Although the slopes at the corners are now zero, the effect of having non-zero slopes can be achieved by properly choosing the maximum allowable slope used by the slope control function. Figure 3.18 depicts the edge model applied to a step size of one with the zero-crossing located at

the half point between the corner samples. The edge is shown for various choices of the maximum slope. Note how the curves are able to bend sharply as the slope is increased, allowing the transition at the edge to be maintained without blurring.

3.2.3 Improved Algorithm Parameters

The choice of the algorithm's two parameters will determine the quality of the HR estimate. The first parameter is the maximum allowable slope S of the edge model, and the second is the regularization parameter α.

The choice of S affects the sharpness of edges in the HR estimate. Small S will result in more gradual edges, which for some images this may cause blurring. Large S will make edges sharp. However, this may also introduce blocking artifacts in images with many large and highly localized edges. This is due to the fact that the edge curve approaches an ideal step-edge as S increases.

The effect of α is as discussed in Section 3.1.1. Note that it is possible to use the iterations to soften blocking resulting from large S.

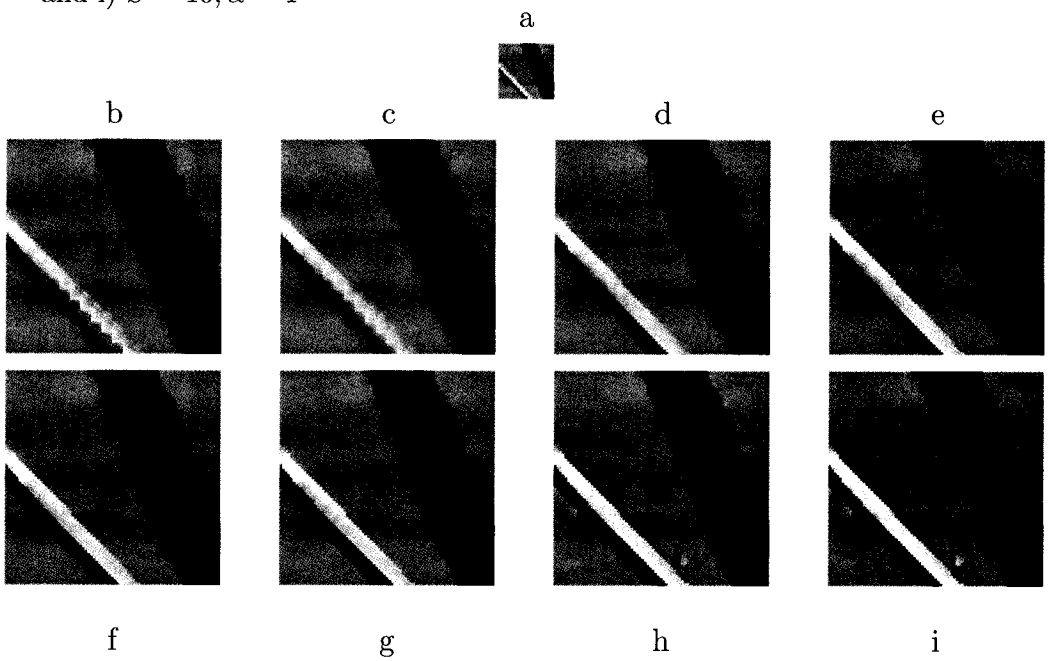
The images in Figure 3.19 illustrate the effect of the parameters. The same test image was used in order to provide a better comparison of the effects of varying both parameters.

The original LR image is shown in part a) of Figure 3.19. Note that the HR estimates b) to c) were obtained with 1 iteration, where as f) to i) were obtained with 10 iterations.

Images b) to e) show the effect of ignoring α (by performing only 1 iteration), while allowing S to vary. Note how the increase in S sharpens the rendering at the edges. This is caused by allowing the edge curve to make use of larger slopes at the zero-crossing.

Images f) to i) show the effect of setting $S = 10$, while allowing α to vary. Observe how increasing α introduces artifacts and a slight ringing around edges. The ringing is caused by use of the moving average sensor model.

Fig. 3.19 Effects of Varying the Parameters of the Improved Edge-Directed Algorithm (16× Surface Magnification): a) Original LR image, HR estimate using b) $S = 0.1, \alpha = 0.75$, c) $S = 1, \alpha = 0.75$, d) $S = 5, \alpha = 0.75$, e) $S = 10, \alpha = 0.75$, f) $S = 10, \alpha = 0$, g) $S = 10, \alpha = 0.25$, h) $S = 10, \alpha = 0.75$ and i) $S = 10, \alpha = 1$



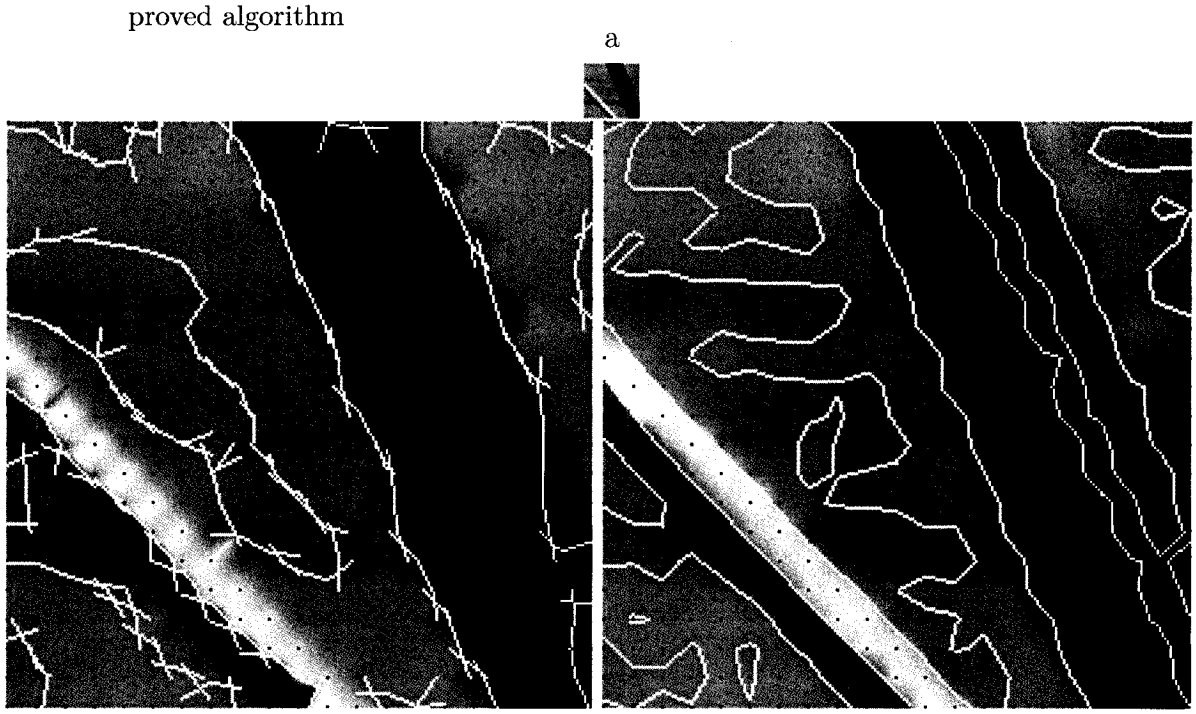
3.2.4 Improved Algorithm Observations

Results from the implementation of the improved edge-directed algorithm show that the problems discussed in Section 3.1.2 have been resolved. This is shown in Figure 3.20, where the regularly spaced black samples indicate the positions of the original LR samples and the white lines are the estimated edges.

Clearly, the proposed sub-pixel edge estimation stage has greatly improved the accuracy of edge localization. This is seen by comparing the bottom left and right images in Figure 3.20. Note how edges in the right image are no longer shifted away from the correct position. Also, LR unit cells are now being properly classified. There are no unit cells with only one zero-crossing.

Further examples of the two edge-directed algorithms will be provided in Chapter 5.

Fig. 3.20 Comparison of the Original and Improved Edge-Directed Algorithms (100× Surface Magnification): a) Original LR image, HR estimate overlayed with estimated edge mapping using b) Original algorithm, c) Improved algorithm



Chapter 4

Statistical Algorithm

This chapter begins by providing a brief overview of the operation of the selected statistical algorithm, as described by the authors of [29]. Following the description, certain improvements to the algorithm will be proposed.

For a detailed description of the implementations, please refer to Appendix C, for the statistical algorithm, and D, for the improved statistical algorithm.

4.1 Algorithm Description

The primary difficulty in image magnification is that there is no unique HR estimate for any given original LR image. In fact, there are an infinite number of possible solutions. The idea behind statistical image magnification is to construct a statistical model which best describes the *a priori* information about the HR image being estimated. The choice of this model will determine the properties of the solution. For example, the model can be designed to give the magnified image an overall smoothness, while keeping edges sharp. The statistical approach also allows integrating noise models into the algorithm.

The statistical algorithm of [29] formulates image magnification as Bayesian estimation. The goal is to determine the most likely HR image that meets the requirements of the *a priori* knowledge and would have resulted in the given LR image after being decimated

by a moving average filter. The estimated HR image is defined as,

$$\mathbf{HR}_{\min} = \arg\min_{\mathbf{HR}}\{-L(\mathbf{HR}|\mathbf{LR})\} \tag{4.1}$$

where,

$$\begin{aligned} L(\mathbf{HR}|\mathbf{LR}) &= \ln p(\mathbf{HR}|\mathbf{LR}) && (4.2)\\ &= \ln p(\mathbf{LR}|\mathbf{HR}) + \ln p(\mathbf{HR}) - \ln p(\mathbf{LR}) && (4.3) \end{aligned}$$

is the likelihood that an estimated HR image would result from a given LR image. $p(\cdot)$ denotes the probability density of its argument.

In Equation 4.2, the term $p(\mathbf{HR}|\mathbf{LR})$ is the conditional probability density function that gives the probability that a particular HR image resulted from the given LR image. According to probability theory, this term can be re-written as

$$p(\mathbf{HR}|\mathbf{LR}) = \frac{p(\mathbf{LR}|\mathbf{HR})p(\mathbf{HR})}{p(\mathbf{LR})}. \tag{4.4}$$

In Equation 4.3, the term $P(\mathbf{LR}|\mathbf{HR})$ represents the likelihood that a decimated version of the estimated HR image could produce the given LR image. The term $P(\mathbf{HR})$ represents the degree to which the estimated HR image meets the criteria established by the *a priori* knowledge. Note that since the term $P(\mathbf{LR})$ does not depend on the HR image being estimated, it will be constant throughout the minimization. Therefore, this term will not affect the estimation process.

The authors of [29] assume that the original LR image will be corrupted by white Gaussian noise. This defines the relationship between the LR and estimated HR images to be

$$p(\mathbf{LR}|\mathbf{HR}) = \frac{1}{(2\pi\sigma^2)^{\frac{X_{LR}Y_{LR}}{2}}} \exp\left(-\frac{\|\mathbf{LR} - \mathbf{D}(\mathbf{HR})\|^2}{2\sigma^2}\right). \tag{4.5}$$

The *a priori* information about the estimated HR image is modelled by the Gibbs-

Markov random field such that

$$p(\mathbf{HR}) = \frac{1}{Z}\exp\left(-\frac{1}{\beta}\Omega[\mathbf{HR}, T]\right). \tag{4.6}$$

Substituting the right-hand side of Equations 4.5 and 4.6 into Equation 4.3 and evaluating the logarithms results in

$$L(\mathbf{HR}|\mathbf{LR}) = -\frac{X_{LR}Y_{LR}}{2}\ln(2\pi\sigma^2) - \frac{1}{2\sigma^2}\|\mathbf{LR} - \mathbf{D}(\mathbf{HR})\|^2 - \ln(Z) - \frac{1}{\beta}\Omega[\mathbf{HR}, T] \tag{4.7}$$

As before, the constant terms in Equation 4.7 will not contribute to the minimization.

From the above, the estimation problem reduces to the following minimization:

$$\mathbf{HR}_{\min} = \arg\min_{\mathbf{HR}}\{M[\mathbf{HR}, T, \lambda]\} \tag{4.8}$$

where,

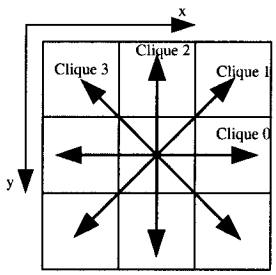
$$M[\mathbf{HR}, T, \lambda] = \Omega[\mathbf{HR}, T] + \lambda\|\mathbf{LR} - \mathbf{D}(\mathbf{HR})\|^2. \tag{4.9}$$

Let X_{LR} and Y_{LR} be the horizontal and vertical dimensions of the LR image, respectively. Similarly, X_{HR} and Y_{HR} are the dimensions of the HR image. Note that in the above, **LR** is a $X_{\mathrm{LR}}Y_{\mathrm{LR}} \times 1$ column vector representation of the LR image, **HR** is a $X_{\mathrm{HR}}Y_{\mathrm{HR}} \times 1$ column vector representation of the HR image and **D** is a decimation operator in the form of an $X_{\mathrm{LR}}Y_{\mathrm{LR}} \times X_{\mathrm{HR}}Y_{\mathrm{HR}}$ matrix, which reduces the HR estimate to the dimensions of the LR image by performing an area averaging of the HR samples contained in an LR unit cell. The single constant λ in Equation 4.9 is the synthesis of the constants $\frac{1}{2\sigma^2}$ and $\frac{1}{\beta}$ from Equations 4.5 and 4.6, respectively. The authors of [29] rearranged and merged the two in order to eliminate a extra parameter to the algorithm. Also, note that contrary to tradition, λ is applied to the data fidelity term instead of the smoothing term. It is believed that this choice was made in order to allow controlling the smoothing solely with T.

The objective function $M[\mathbf{HR}, T, \lambda]$ has two components: i) the smoothing term

$\Omega[\mathbf{HR}, T]$, and ii) the data fidelity term $\|\mathbf{LR} - \mathbf{D}(\mathbf{HR})\|^2$.

Fig. 4.1 Cliques at a Sample



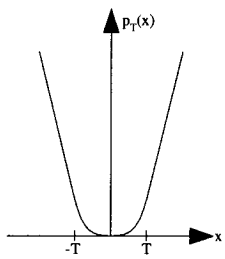
The smoothing term constrains the sharpness of the image. A second-order derivative operator is used as a measure of how smooth an image is at a sample. The decision to use a second-order operator is motivated by the fact that it is zero in smooth regions and non-zero in regions containing discontinuities. Therefore, it provides a simple mechanism for measuring the smoothness of a sample with respect to it's neighbors. For each sample, a measure is taken in each of the four directions shown in Figure 4.1: horizontal (clique 0), anti-diagonal (clique 1), vertical (clique 2) and diagonal (clique 3). A clique is defined as a group of samples used to measure some local property of an image. In this case, the cliques measure the smoothness by using three adjacent samples along one of the four directions shown in Figure 4.1. Since only the eight nearest neighbors of a sample are used, the four cliques are said to be defined on a second-order neighborhood. Smooth regions possess small second-order derivatives, whereas regions containing discontinuities will result in larger values. Refer to Section C.1.1 of Appendix C for further details on the smoothness measures.

A potential function is then used to transform the smoothness measures. The goal of this transformation is to obtain a new measure which simplifies the optimization. When choosing a potential function, it is desired that it transforms the smoothness measures in such a way as to create greater separation between values. This greater separation allows the optimization procedure to target those regions which will have the greatest overall

effect on reducing the objective function.

Traditionally, the quadratic function is chosen as the potential, which results in a least-squares estimation of the HR image. Unfortunately, the quadratic function tends to cause excessive blurring by severely penalizing discontinuities. Although it is desirable to create separations between smoothness measures, the quadratic is unable to be adapted to permit various degrees of discontinuities. Essentially, any discontinuity will be smoothed in order to reduce the objective function, regardless of its perceptual significance. Clearly, a better choice for the potential function should be made.

Fig. 4.2 Huber Function



This new potential function should continue to ensure that smooth regions with small second-order derivatives have a weak influence on the global objective. However, regions containing discontinuities should be limited in some way as to reduce their influence, and hence the blurring. From previous research on discontinuity preserving potential functions performed in [39], the authors of [29] chose the Huber function (illustrated in Figure 4.2), which is piece-wise smooth and possesses a continuous first order derivative. This function is defined as

$$\rho_T(x) = \begin{cases} x^2 & , |x| \leq T \\ T^2 + 2T(|x| - T), & |x| > T \end{cases} . \tag{4.10}$$

Note that this function becomes purely quadratic as the threshold T becomes arbitrarily large. This means that large values of T will reduce the Huber function to a least-squares solution for the HR estimate.

The strength of the Huber function lies in the ability to select the threshold T. This threshold can be chosen such that discontinuities are not severely penalized, thereby reducing excessive blurring. The threshold serves as a control mechanism which allows the final estimated solution to differ from the least-squares solution, while at the same time remaining relatively similar to it. Expanding the smoothing term gives,

$$\Omega[\mathbf{HR}, T] = \sum_{x=0}^{X_{\mathrm{HR}}-1} \sum_{y=0}^{Y_{\mathrm{HR}}-1} \sum_{c=0}^{3} \rho_T(\mathbf{d}_{x,y,c}^t \mathbf{HR}), \tag{4.11}$$

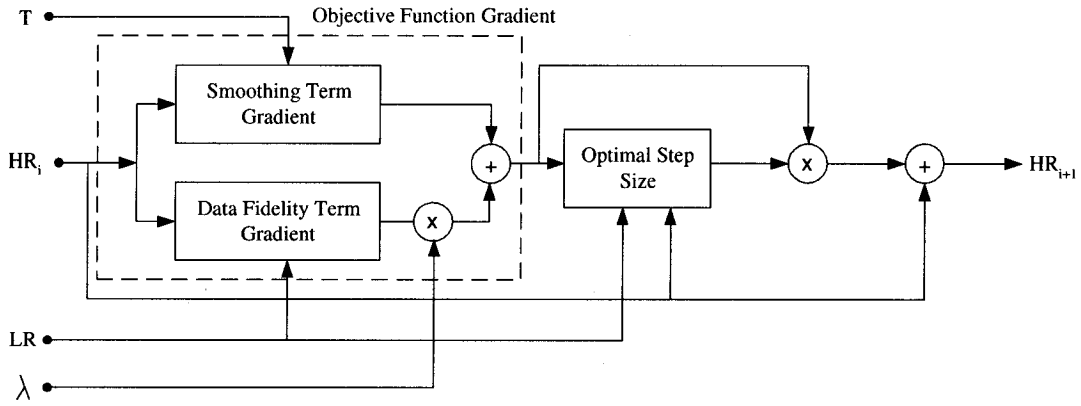
where $\mathbf{d}_{x,y,c}$ are the directional smoothness measures for the four cliques.

The data fidelity term constrains the solution space by ensuring that the HR estimate does not deviate radically from the original LR image. This is accomplished by making certain that a decimated version of the HR estimate remains close to the original LR image. The degree of the allowable deviation is determined by the regularization parameter λ. The importance of the data fidelity term lies in the need to maintain the information of the original image. Without data fidelity, the algorithm would allow the smoothing term to seek the minimum solution, which would result in an image of constant and uniform intensity. The choice of λ must be made such that an acceptable level of data fidelity is maintained while not overly restricting the smoothing term from removing any undesirable artifacts in the HR estimate or from improving desired features such as discontinuities. The decimation is performed using a moving average filter, followed by down-sampling. This filter was selected to model image acquisition performed using modern digital sensors such as CCD or CMOS devices.

The optimization of the objective function is accomplished using the gradient descent technique. Using this technique, the implementation of the algorithm is segmented into three components: i) gradient of the objective function, ii) optimal step size, and iii) correction. A block diagram of the implementation is given in Figure 4.3.

The computation of the gradient of the objective function requires the gradients of the smoothing and data fidelity terms. The gradients are computed with respect to each

Fig. 4.3 Block Diagram of the Bayesian Optimization



sample in the HR estimate to generate vectors of the same dimensions as the HR estimate. The gradient vector determines the optimal direction in which to change the HR estimate. This means that the estimated sample's grey-levels will be increased or decreased by some amount in order to force them closer to their optimal values. The gradient at every sample is computed solely from it's neighboring samples in the current HR estimate. Refer to Sections C.1.2, C.1.3 and C.1.4 of Appendix C for further details on the gradient of the objective function.

The step size is the constant factor which determines the magnitude of the change of the HR estimate in the direction of the gradient. The choice of the step size is crucial in determining the convergence of the algorithm. A small step will converge slowly, whereas a large step may overshoot the estimate. The solution used by [29] computes the optimal step size at each iteration. This is done by choosing the step size that minimizes the objective function with respect to the gradient corrected HR estimate. Using this approach ensures that the change at each iteration will be uniquely adapted for the gradient at that iteration. Refer to Section C.1.5 of Appendix C for further details on the optimal step size.

The correction is simply the modification of the HR estimate by the gradient vector weighted with the optimal step size to produce the next best estimate. Samples in the

HR estimate are updated simultaneously by their respective gradients. The correction is only performed provided that the squared magnitude of the correction term is greater than some specified value. Otherwise, the algorithm terminates. Refer to Section C.1.6 of Appendix C for further details on correction.

Naturally, the algorithm will require a starting point of some sort. In [29], the authors chose the zero-order hold expansion of the LR image as the initial condition. This choice of initial condition is reasonable given that the sensor model is the area averaging. The initial condition does not affect the result provided that the algorithm is allowed to converge to zero. However, it should be noted that halting the iterations before convergence results in a sharper HR estimate.

4.1.1 Algorithm Parameters

The choice of the algorithm's two parameters will determine the quality of the HR estimate. The first parameter is the threshold T of the Huber function, and the second is the regularization parameter λ.

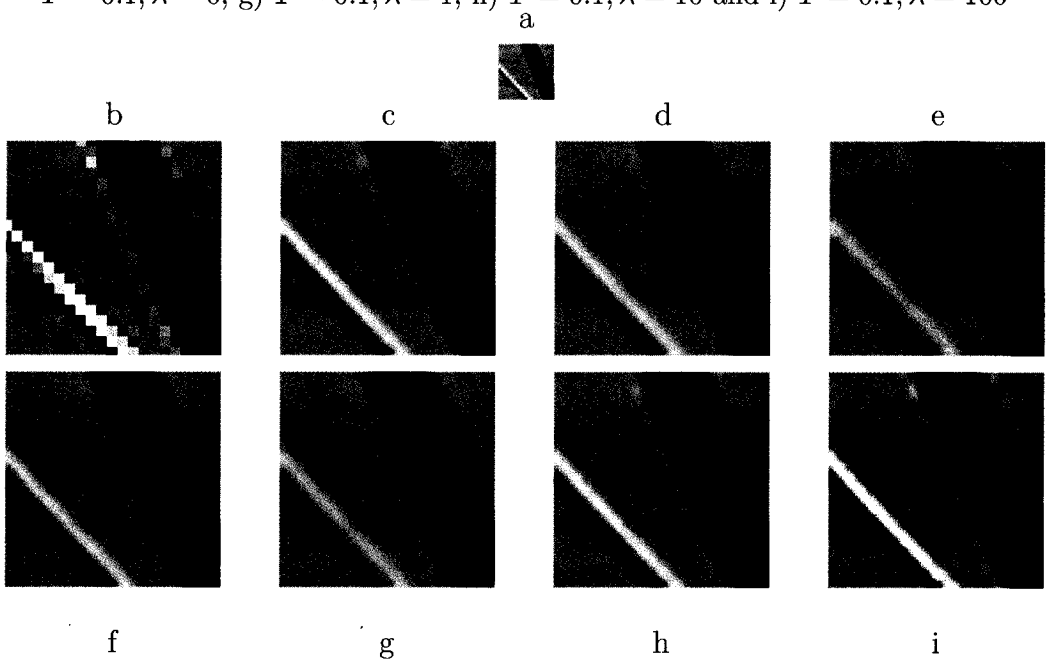
The choice of T affects the amount of smoothing performed. For a fixed λ, increasing T will increase the amount of smoothing performed at each iteration.

The choice of λ determines the degree of data fidelity. Since the sensor model is the moving average, it should be clear that making λ large will result in blocking artifacts. In the limit, λ will cause the HR estimate to remain in the zero-order hold state. Note, that although not an explicit parameter, the maximum number of iterations can also be used to control convergence.

The images in Figure 4.4 illustrate the effect of the parameters. The same test image was used in order to provide a better comparison of the effects of varying both parameters.

The original LR image is shown in part a) of Figure 4.4. Note that all HR estimates were performed for 100 iterations. This number was chosen in order to allow all estimates to converge fully. Further iteration beyond this point would not result in improvements. Instead, the algorithm would continue oscillating around the solution.

Fig. 4.4 Effects of Varying the Parameters of the Statistical Algorithm (16× Surface Magnification): a) Original LR image, HR estimate using b) $T = 0, \lambda = 0$, c) $T = 0.05, \lambda = 0$, d) $T = 0.1, \lambda = 0$, e) $T = 1, \lambda = 0$, f) $T = 0.1, \lambda = 0$, g) $T = 0.1, \lambda = 1$, h) $T = 0.1, \lambda = 10$ and i) $T = 0.1, \lambda = 100$



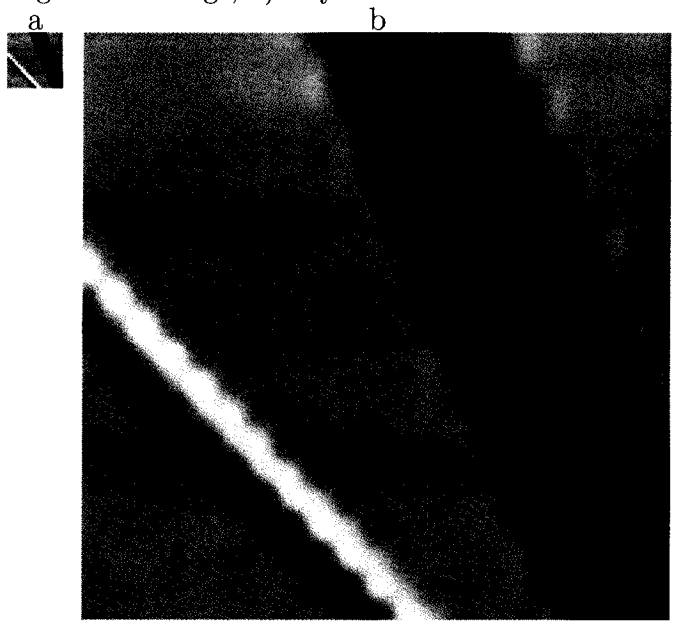
Images b) to e) show the effect of setting $\lambda = 0$, while allowing T to vary. Clearly, the contribution of T is responsible for smoothing. Note how the HR estimate becomes gradually more blurred as T is increased.

Images f) to i) show the effect of setting $T = 0.1$, while allowing λ to vary. The contribution of λ is responsible for ensuring data fidelity. Note how increasing the data fidelity begins to introduce ringing. This effect results from the use of the moving average filter as the sensor model.

4.1.2 Algorithm Observations

From the implementation of the statistical algorithm of [29], it was observed that despite the use of the Huber function to reduce the amount of smoothing, discontinuities were still being smoothed. Figure 4.5 clearly shows the smoothing effect near edges.

Fig. 4.5 Difficulties with the Statistical Algorithm (100× Surface Magnification): a) Original LR image, b) Bayesian HR estimate



In addition to the smoothing, blocking artifacts are visible along discontinuities. These are caused by the zero-order hold of the LR image, which is used as an initial condition. The smoothing is clearly not sufficient to remove them. Also, the fact that the sensor model is a moving average places a limit on the degree to which the blocking artifacts can be removed. This is due to the fact that the moving average would force the HR estimate towards the zero-order hold solution, and not a solution with straighter lines.

A first attempt at solving this problem focused on using smaller thresholds for the Huber function. However, it was discovered that when the threshold became too small, the algorithm becomes numerically unstable. To solve this, it was decided to approximate the step size by the optimal step size obtained by allowing the Huber function's threshold to be infinite (i.e. a least-squares solution). This approach eliminates the dependency on the threshold. However, since the step size is no longer optimal, the algorithm will require more iterations before converging.

Although the redefinition of the step size did help reduce some of the blurring, the

solution was still lacking. As the threshold was reduced, blocking artifacts began to be introduced. Eventually, the threshold would reach a state in which the optimal solution was the zero-order hold initial condition. The only way to reduce the blocking artifacts was to reduce the regularization parameter, which reduces data fidelity. This in turn allows for more smoothing. Essentially, there was a trade-off to be made between smoothness and data fidelity. It became clear that the Huber function did not provide sufficient control over the amount of smoothing applied to each region of the HR estimate. Naturally, the question arose as to what are the properties that would allow such a control mechanism to find the ideal trade-off between smoothness and data fidelity, while at the same time maintaining sharpness at discontinuities? This question led to the improvements presented in the next section.

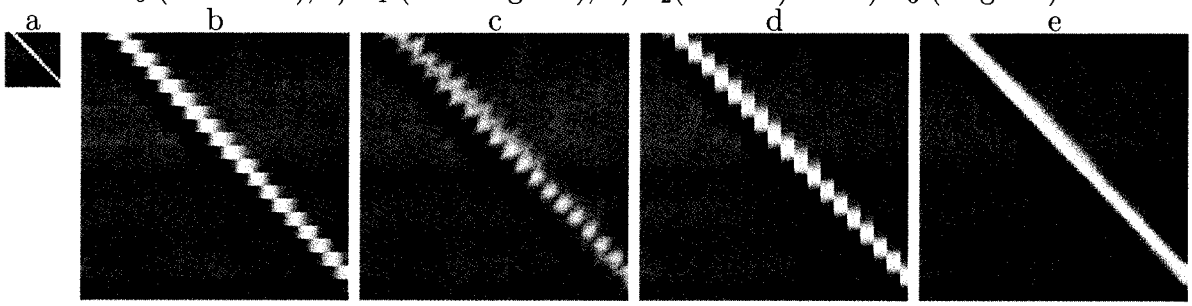
4.2 Improved Algorithm

To answer the question of Section 4.1.2, it was necessary to return to the basic least-squares algorithm upon which [29] is based. Analysis of the least-squares algorithm revealed that the primary factor responsible for the blurring of discontinuities was the generality with which all four clique smoothness measures are applied to every sample in the HR estimate. To demonstrate this idea, simple tests were conducted where four HR estimates of the same LR image were obtained using only one of the four directional smoothness measures during each test (the contribution of the others were set to zero). This gives four HR estimates which clearly show the contribution of each of the four cliques to the overall smoothing of the HR estimate. Figure 4.6 depicts the four directional smoothing components for an image containing a diagonally oriented discontinuity. The original image, a), was interpolated by a factor of 5 in each dimension, resulting in a 25× surface increase.

Note that the clique smoothness measures used by [29] are not weighted the same for each clique. The two diagonal cliques have half the weight of the horizontal and diagonal cliques. In order to obtain an accurate idea of the effect of each clique, it was decided to

give them all the same weight. This way, an equal amount of smoothing will be performed by each clique.

Fig. 4.6 Effects of Using One Clique on the HR Estimate (25× Surface Magnification): a) Original LR image, HR estimate using smoothness measure b) $\mathbf{d}_0$ (horizontal), c) $\mathbf{d}_1$ (anti-diagonal), d) $\mathbf{d}_2$(vertical) and e) $\mathbf{d}_3$ (diagonal)



For the particular image in Figure 4.6, it is clear that of the four estimates, the fourth (image e) gives the most accurate estimation near the discontinuity. This is due to the fact that the other directional components are smoothing across the discontinuity and not along it. In particular, observe the effect of the smoothing term that is perpendicular to the discontinuity (image c). By smoothing directly across the discontinuity, this smoothing term creates jagged artifacts. For the case of this particular image, it is clear that only the fourth smoothing component should be allowed to affect regions near the discontinuity, whereas for regions beyond it, all four should be applied.

As a result of these tests, it became clear that the control mechanism should in some way permit or block each of the four directional smoothing components based on the presence of local discontinuities. Smoothing would therefore only be conducted along the discontinuity and not across it.

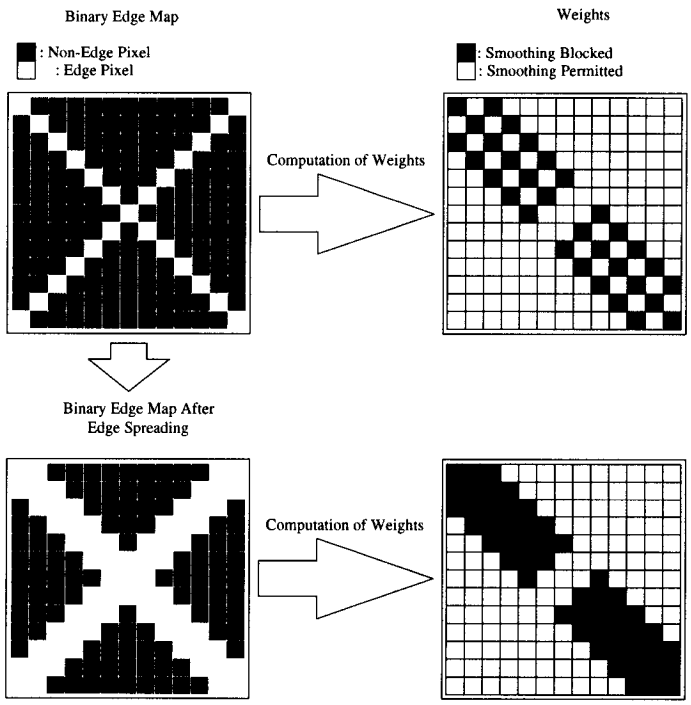
A simple way of achieving such a control mechanism is to introduce a set of weights which would scale each of the four smoothness measures taken at every sample in the HR estimate. These weights would be designed specifically for the direction in which each of the smoothing terms operates.

An obvious choice in designing the weights would be to base them on an estimate of the HR edge map. The edge map would provide a binary estimate of the location

of the discontinuities in the HR image, where a value of one indicates the presence of a discontinuity. From these locations, a simple scheme can be applied to determine the weights for each directional smoothing term.

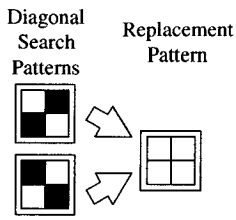
The idea is to eliminate the smoothing only at samples for which the smoothing term is crossing a discontinuity. If a smoothing term is applied along a discontinuity, smoothing should be permitted. To achieve this, note that the directional smoothing operators are all second-order operators. Filtering the binary edge map with a particular operator will result in non-zero values only for those samples whose clique extends to include a discontinuity. Samples whose clique follows a discontinuity will result in zero. Therefore, taking the absolute value of the filtered edge map and then applying a threshold such that anything greater than zero is set to one, will result in the location of all samples that should not be smoothed using that particular operator. The weights are therefore the logical inverse of such binary maps.

Fig. 4.7 Spreading Edges for Diagonal Operators



A problem arises in the case of the two diagonal measures. Figure 4.7 illustrates a binary edge map with two crossing diagonal edges. When applying either of the two diagonal smoothing measures, certain holes appear that will permit smoothing across discontinuities. To ensure that the holes are properly dealt with, they should be filled in by spreading the edge. The spreading of the edge map can be accomplished by the application of a simple morphological operator that searches the edge map for groups of four samples with the two patterns shown in Figure 4.8. If a match is found, the operator fills in the holes.

Fig. 4.8 Morphological Operator for Spreading Diagonal Edges



The determination of the weights can be summarized as,

$$w_c[x,y] = 1 - e_c[x,y] \tag{4.12}$$

where, the subscript c denotes the clique and

$$e_0[x,y] = |d_0[x,y] * e[x,y]| > 0, \tag{4.13}$$

$$e_1[x,y] = |d_1[x,y] * s(e[x,y])| > 0, \tag{4.14}$$

$$e_2[x,y] = |d_2[x,y] * e[x,y]| > 0, \tag{4.15}$$

$$e_3[x,y] = |d_3[x,y] * s(e[x,y])| > 0. \tag{4.16}$$

In the above, $e[x,y]$ denotes the binary estimate of the HR edge map, $*$ denotes the convolution operator, $d_i[x,y]$ denotes the 3×3 filter for i^{th} directional smoothness measure

and $s(\cdot)$ is the morphological operator used for spreading edges.

Fig. 4.9 Determination of the Weights

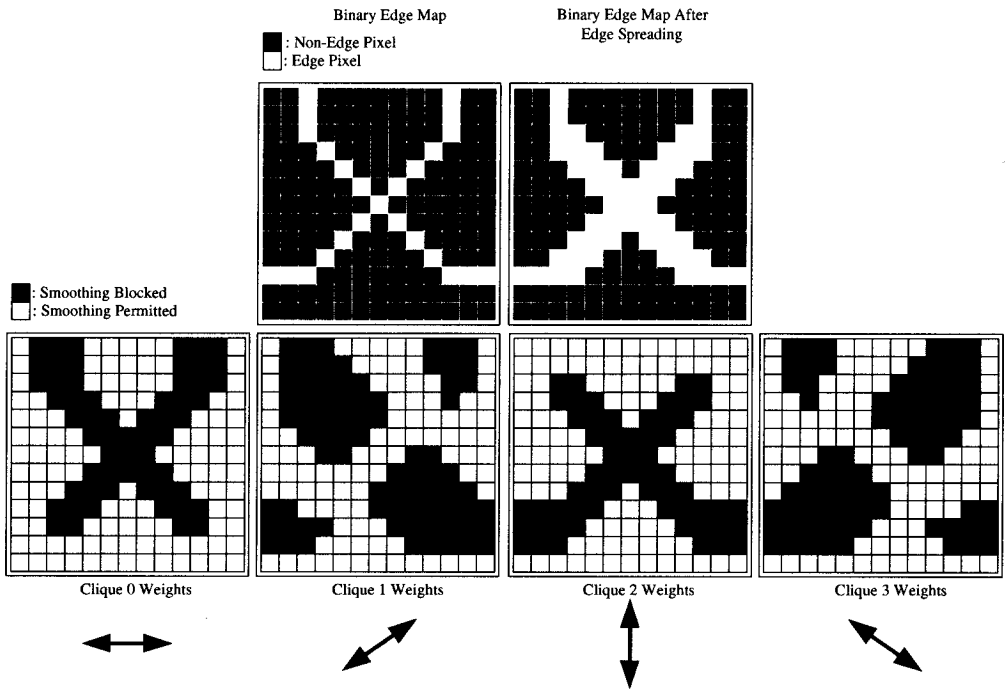


Figure 4.9 illustrates the weights for a simple binary edge map. Note how the weights permit smoothing along discontinuities that are parallel to the direction of the smoothness measure, and block smoothing across discontinuities that are perpendicular to the smoothness measure. Also, due to the width of the smoothness measures, every discontinuity has a region of at least a single sample on either side of it that will be smoothed using the same measure that is used on the samples that lie directly on the discontinuity. This buffer zone around the discontinuities adds robustness to the algorithm by allowing for the correction of one sample deviations in the estimated edge map. Basically, a three sample wide area is smoothed using a single measure, which will maintain sharpness of discontinuities despite errors in the estimated edge map.

Since the edge map is binary, the weights will never allow smoothing across discon-

tinuities. In order to relax this constraint and add flexibility, the single regularization parameter, λ, used by [29] will no longer be applied to the data fidelity term. Instead, two independent parameters, λ_1 and λ_2, will be applied to the smoothing term such that the weight in smooth regions is λ_1 and the weight in regions of discontinuity is λ_2. This allows for greater control over the degree of smoothing that is performed. Note that the two new parameters λ_1 and λ_2 are fundamentally different from the single parameter λ used in [29]. The new parameters are applied to the smoothing term, where as λ is applied to the data fidelity term. For the original algorithm, changes in λ affect the degree to which a down-sampled version of the HR estimate matches the original LR image. In the improved algorithm, changes in λ_1 and λ_2 affect the amount of smoothing performed in their respective regions. The parameters of the original and new algorithms are not related, hence they will affect the convergence of each algorithm in different ways.

In addition, the improved algorithm will no longer use the Huber function. Instead, the clique potential function will be a simple quadratic. This change was made following experimentation with the Huber function. It was found that the threshold of the Huber function needed to be large in order to avoid blocking artifacts. The combination of the weights and the Huber function's further reduction of the smoothness measures was causing the solution to be reached too soon. The large threshold was such that only the quadratic segment of the Huber function was being applied to the smoothness measures.

From the above, the objective function for the algorithm becomes,

$$M[\mathbf{HR}, \lambda_1, \lambda_2] = \Omega[\mathbf{HR}, \lambda_1, \lambda_2] + \|\mathbf{LR} - \mathbf{D}(\mathbf{HR})\|^2 \tag{4.17}$$

where,

$$\Omega[\mathbf{HR}, \lambda_1, \lambda_2] = \sum_{x=0}^{X_{HR}-1} \sum_{y=0}^{Y_{HR}-1} \sum_{c=0}^{3} w_c[x, y, \lambda_1, \lambda_2](\mathbf{d}_{x,y,c}^t \mathbf{HR})^2, \tag{4.18}$$

and

$$w_c[x, y, \lambda_1, \lambda_2] = (\lambda_2 - \lambda_1)e_c[x, y] + \lambda_1. \tag{4.19}$$

The estimation of the HR edge map will be performed using the improved sub-pixel edge estimator of Section 3.2.1. However, the estimator does not generate a binary edge map. Therefore, an additional step must be included to convert the zero-crossing structures into a binary map. Since the edges are approximated by piece-wise linear segments, this step need only quantize the straight line segments joining adjacent zero-crossings to the HR grid within every LR unit cell.

4.2.1 Algorithm Parameters

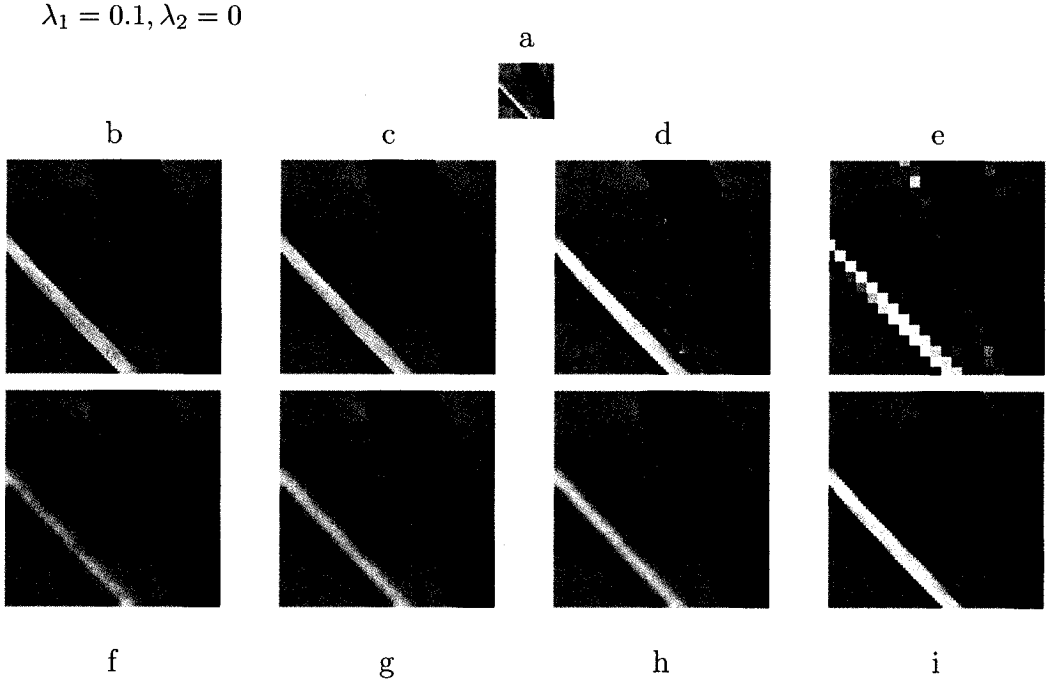
The choice of the algorithm's two parameters will determine the quality of the HR estimate. The first parameter, λ_1, is the weight applied to smooth regions, and the second, λ_2, is the weight applied to regions containing discontinuities.

The choice of λ_1 affects the amount of smoothing performed in all regions that do not cross any discontinuities. If λ_1 is set to zero, then no smoothing will be performed. This means that samples not located near a discontinuity will maintain the same value given to it by the zero-order hold initial condition, regardless of the number of iterations. As λ_1 increases, more smoothing will be performed by increasing the contribution of the smooth regions to the global objective.

The choice of λ_2 affects the amount of smoothing performed across discontinuities. If λ_2 is set to zero, then no smoothing across discontinuities will be performed. As λ_2 increases, discontinuities will become increasingly blurred. It should be noted that using $\lambda_2 = 0$ for a small magnification factor will result in blocking. The problem is that the resolution of the estimated edge mapping will be low. This means that areas with high edge content will be predominantly smoothed using the λ_2 parameter instead of λ_1. For images with high edge content, the blocking will be persistent across the entire HR estimate. The solution is to allow smoothing to be performed across edges or to use a greater magnification factor.

The images in Figure 4.10 illustrate the effect of the parameters. The same test image was used in order to provide a better comparison of the effects of varying both parameters.

Fig. 4.10 Effects of Varying the Parameters of the Statistical Algorithm (16× Surface Magnification): a) Original LR image, HR estimate using b) $\lambda_1 = 1, \lambda_2 = 0$, c) $\lambda_1 = 0.5, \lambda_2 = 0$, d) $\lambda_1 = 0.01, \lambda_2 = 0$, e) $\lambda_1 = 0, \lambda_2 = 0$, f) $\lambda_1 = 0.1, \lambda_2 = 0.5$, g) $\lambda_1 = 0.1, \lambda_2 = 0.1$, h) $\lambda_1 = 0.1, \lambda_2 = 0.05$ and i) $\lambda_1 = 0.1, \lambda_2 = 0$



The original LR image is shown in part a) of Figure 4.10. Note that all HR estimates were performed for 100 iterations.

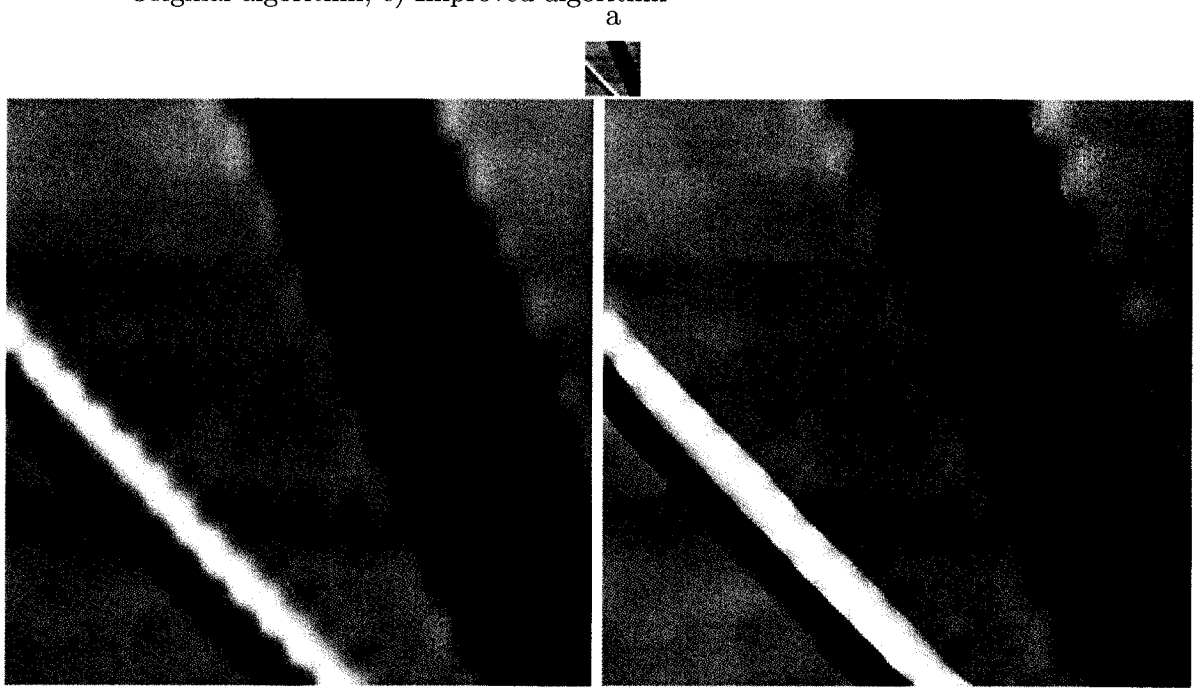
Images b) to e) show the effect of setting $\lambda_2 = 0$, while allowing λ_1 to vary. Note how the reduction of λ_1 is responsible for introducing blocking artifacts. This is due to insufficient smoothing.

Images f) to i) show the effect of setting $\lambda_1 = 0.1$, while allowing λ_2 to vary. Observe how decreasing λ_2 has the effect of sharpening of discontinuities.

4.2.2 Improved Algorithm Observations

From the implementation of the improved statistical algorithm, it can be observed that smoothing at discontinuities has been significantly reduced. The image on the bottom-right of Figure 4.11 shows the improvements compared to the image on the bottom-left.

Fig. 4.11 Comparison of the Original and Improved Statistical Algorithms (100× Surface Magnification): a) Original LR image, HR estimate using b) Original algorithm, c) Improved algorithm



Clearly, the decision to perform directional smoothing based on the orientation of the discontinuities in an image has had a positive effect on the overall sharpness of the image. Also, the directional smoothing has helped to remove much of the blocking artifacts along discontinuities. By only allowing samples that lie on the same side of a discontinuity to affect each other, a more uniform spreading of the intensity is achieved. This forces the samples in a blocking artifact to converge to an intensity that is closer to those on the same side of the discontinuity.

Further examples of the two statistical algorithms will be provided in Chapter 5.

Chapter 5

Algorithm Comparison

This chapter provides a comparison of the two selected magnification algorithms of Chapters 3 and 4, as well as their respective improvements. The standard bicubic interpolation is also included as a benchmark.

For consistency, it was decided to use the same set of parameters for all test images. The parameters for each of the four algorithms are shown in Table 5.1.

Table 5.1 Parameters Used for Comparing the Algorithms

Algorithm	Parameters	Reference
Edge-Directed	$\alpha = 0.25$	Section 3.1, Chapter 3
Improved Edge-Directed	$S = 10, \alpha = 0.25$	Section 3.2, Chapter 3
Statistical	$T = 0.05, \lambda = 10$	Section 4.1, Chapter 4
Improved Statistical	$\lambda_1 = 0.1, \lambda_2 = 0$	Section 4.2, Chapter 4

The edge-directed algorithms performed 10 iterations before producing the final HR estimate. The statistical algorithms performed 200 iterations before producing the final HR estimate. The iterations were allowed to run until completion. The choice of iterations was made to allow each algorithm to converge. Since the various images converge differently, the iterations were chosen based on the convergence of the slowest image. The slowest to converge is shown in Figure 5.7.

The above parameters were chosen based on previous experimentation across a variety of test images, many of which are not included in the following comparison. These parameters were found to provide the best subjective results for a magnification factor of 5 in the horizontal and vertical directions (25× surface magnification).

The average time required by each of the algorithms to magnify images of various sizes is given in Table 5.2. Although the implementation of the algorithms was not optimized for speed, the timing has been listed to give a general idea of their relative speeds. The tests were performed on a PC with an AMD Athlon XP 2100+ processor running at 2.0 GHz with 512 Mb of RAM.

Table 5.2 Average Time Required to Magnify Various Sized Images by a Factor of 25× Surface Increase

	Time (seconds)		
Algorithm	10 × 10	50 × 50	100 × 100
Edge-Directed	5.31	177.74	496.56
Improved Edge-Directed	0.66	34.61	408.70
Statistical	3.59	128.31	631.45
Improved Statistical	1.89	53.89	307.30

Throughout the development of the algorithms, noise sensitivity was not explicitly accounted for. If excessive noise in an image is a concern, a de-noising technique should be applied prior to magnification. Since one of the goals of the improved edge detection was to increase sensitivity, it is obvious that the edge detector will be strongly affected by noise. Although the sensitivity to noise can be a significant drawback for certain applications, this decision was made in order to ensure that the utmost care would be taken when finding edges in an image. Effects of weak noise on the edge detection can result in shifted edges, where as strong noise will create false edges in the estimated HR edge map. Note that the following images were magnified without any prior de-noising.

Fig. 5.1 LR Camera Man

Fig. 5.2 25× Bicubic Camera Man

The first LR test image is shown in Figure 5.1. This image was selected due to the variety of edge thicknesses present. Narrow edges are difficult to magnify without introducing blocking, where as thick edges can also be a challenge to ringing. Figure 5.2 is the bicubic interpolation.

Fig. 5.3 25× Edge-Directed Camera Man

Fig. 5.4 25× Improved Edge-Directed Camera Man

Figures 5.3 and 5.4 show the results obtained from the original and improved edge-directed algorithms, respectively. Observe how the improved algorithm is able to eliminate the jagged, saw-tooth artifacts along the handlebar of the camera. This clearly shows how the new edge detection strategy is capable of properly estimating the orientation of edges using the local trends around an edge. Also, note how edges in Figure 5.4 appear sharper and more crisp. This results from the new edge model. By allowing the slope at zero-crossings to be large, edges sharpened.

Unfortunately, the use of straight line segments in estimating edges has a drawback. In particular, observe the shadow under the eye in Figure 5.4. The increased sensitivity of the improved edge-detection is able to find those edges. However, the straight line segments fitted between zero-crossings are unable to bend in order to form a smoother edge, resulting in the sharp point that can be observed in the shadow. Also, the large value chosen for the maximum slope will enforce the sharpness of the corner. These artifacts can be softened by selecting a smaller value for the maximum slope. However, doing so

will also introduce blurring at edges.

Fig. 5.5 25× Statistical Camera Man

Fig. 5.6 25× Improved Statistical Camera Man

Figures 5.5 and 5.6 show the results obtained from the original and improved statistical algorithms, respectively. Observe how blurring is significantly reduced in Figure 5.6. This can be seen particularly in the face, where variations in the skin appear less fogged. Also, ringing at edges is not present in the improved algorithm. This is due to the fact that edges are no longer crossed when performing the smoothing.

Unfortunately, the sharpness of the image also results in certain artifacts. Observe the straightness of the tip of the nose in Figure 5.6. This is caused by the choice of λ_2 and the limitations of the directions used by the four cliques. Since $\lambda_2 = 0$, no smoothing will be allowed across edges. Also, since the cliques are only defined along fixed directions, they can not adapt to any other orientations. This means that only the vertical clique (clique 2 of Figure 4.1) will be used, resulting in the observed straight segment along the tip of the nose. Similarly, the bridge of the nose will only be smoothed by the diagonal clique (clique 3 in Figure 4.1). These artifacts can be smoothed by selecting $\lambda_2 > 0$, which

will allow smoothing across discontinuities. Proper selection of λ_2 will allow for controlled smoothing.

Comparing Figure 5.4 and 5.6 to the bicubic interpolation of Figure 5.2, they both show improvements in sharpness, reduction of blurring and blocking artifacts. Also, comparing Figures 5.4 and 5.6 to each other, it can be seen that the improved statistical algorithm is better able to smooth the effect of the straight line segments used to join zero-crossings when estimating the edges. This is due to the fact that the improved edge-directed algorithm does not perform smoothing. Instead, it is only able to interpolate along edges.

Fig. 5.7 LR Barb's Knee

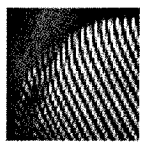
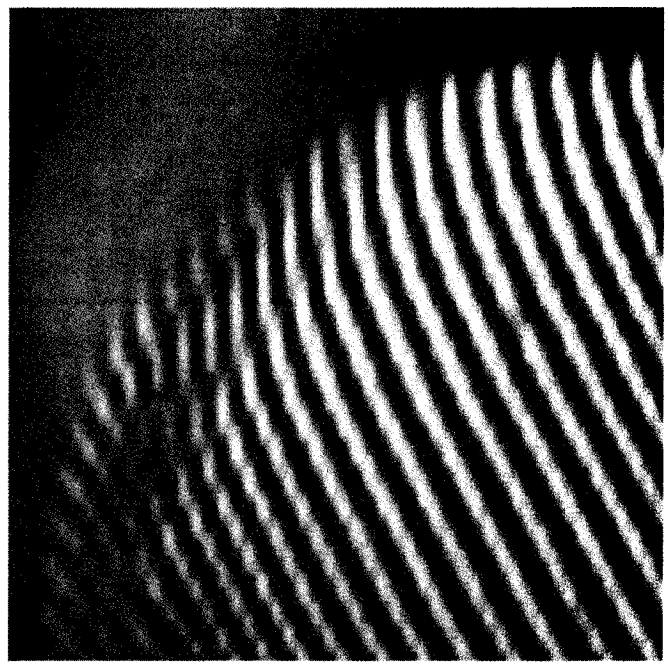


Fig. 5.8 25× Bicubic Barb's Knee



The second LR test image is shown in Figure 5.7. This image was chosen due to it's high frequency content, represented by the closely spaced line patterns. This image is particularly difficult to interpolate without introducing jagged edges. Figure 5.8 is the bicubic interpolation.

Fig. 5.9 25× Edge-Directed Barb's Knee

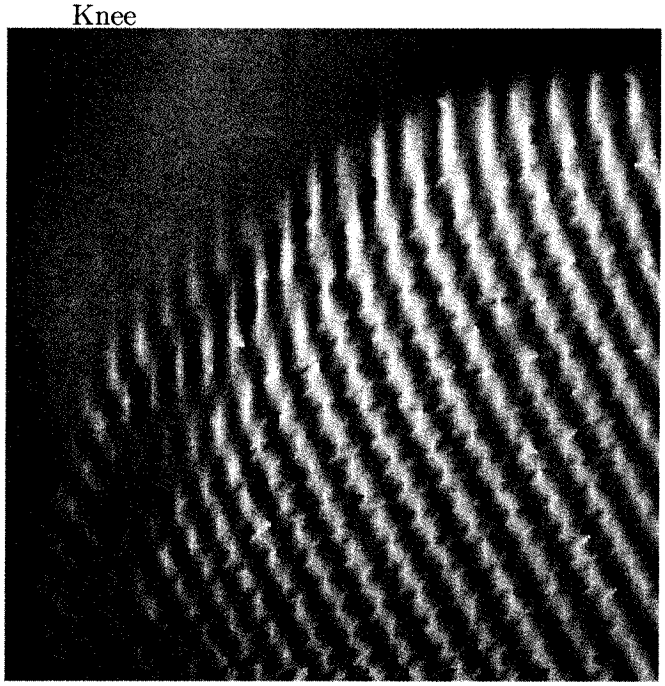
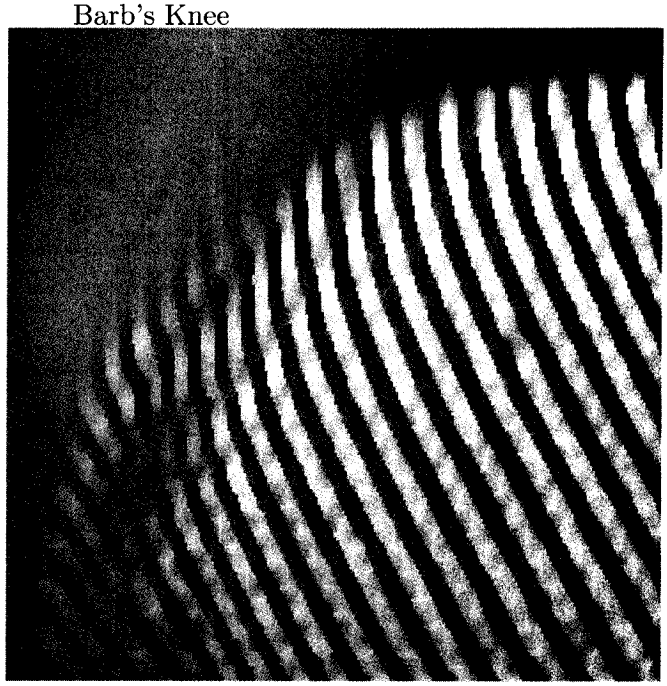


Fig. 5.10 25× Improved Edge-Directed Barb's Knee



Figures 5.9 and 5.10 show the results obtained from the original and improved edge-directed algorithms, respectively. Observe how the improved algorithm produces significantly cleaner edges. This image also serves to highlight the new edge detector's failure to detect weaker edges in the presence of stronger ones. This is seen in the streaks that spread between the darker lines.

Fig. 5.11 25× Statistical Barb's Knee

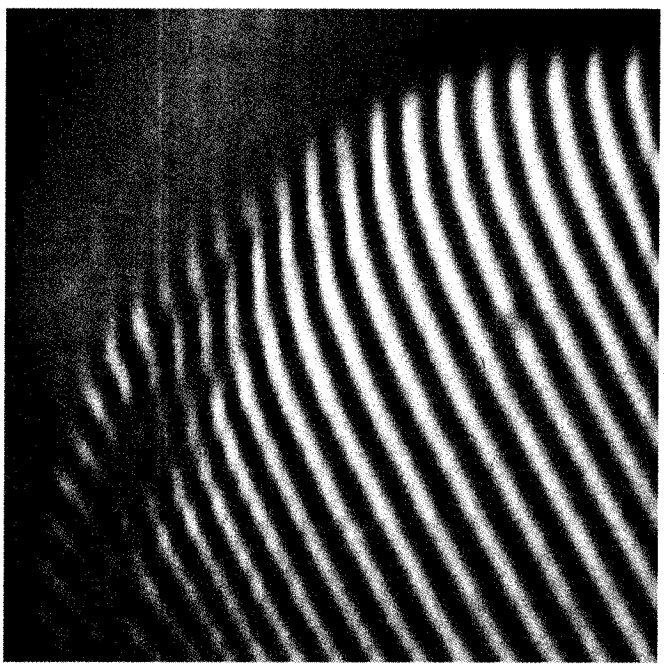
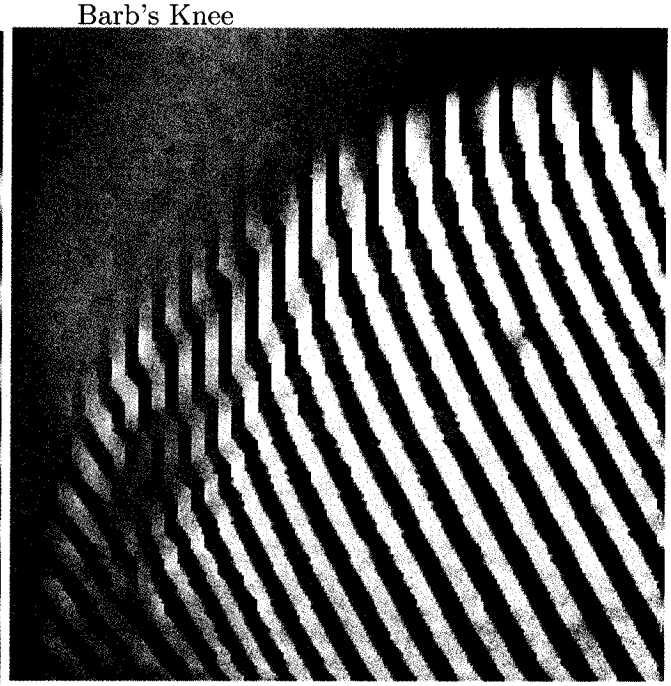


Fig. 5.12 25× Improved Statistical Barb's Knee



Figures 5.11 and 5.12 show the results obtained from the original and improved statistical algorithms, respectively. Observe once again how blurring is significantly reduced in Figure 5.6. Clearly, the contrast at edges is much greater.

Once again, the limited directionality of the four cliques in the improved statistical algorithm can be seen to overly straighten edges that should perhaps be a little more gradual. This can be seen near the top of the knee in Figure 5.12.

Comparing Figure 5.10 and 5.12 to the bicubic interpolation of Figure 5.8, they both show improvements. However, the improved statistical algorithm is clearly superior in it's ability to eliminate blocking and enhance edges. Despite this, the choice of parameters has allowed the improved edge-directed algorithm to give smoother edges than the improved statistical algorithm for this particular image. Although Figure 5.10 does have some blurring, the smoother edges near the top of the knee appear less artificial. This effect can be softened in Figure 5.12 by choosing $\lambda_2 > 0$.

Fig. 5.13 LR Aluminum

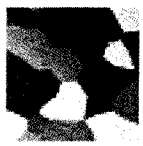
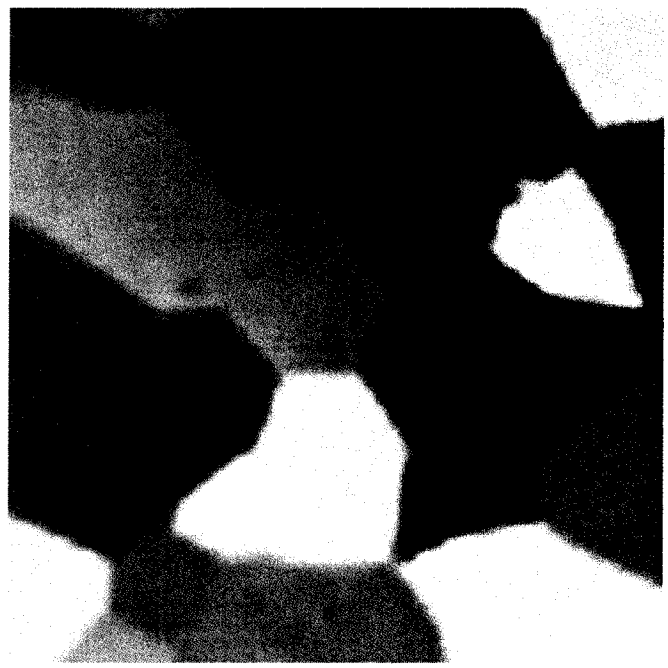


Fig. 5.14 25× Bicubic Aluminum



The third LR test image is shown in Figure 5.13. This image was chosen due to it's wide and relatively constant regions. When edges are too close, the contrast at an edge is lost in the other details in the surrounding area. Using an image whose edges are widely separated by regions of constant intensity will help to demonstrate the contrast enhancing abilities of a particular magnification algorithm. Figure 5.14 is the bicubic interpolation.

Fig. 5.15 25× Edge-Directed Aluminum

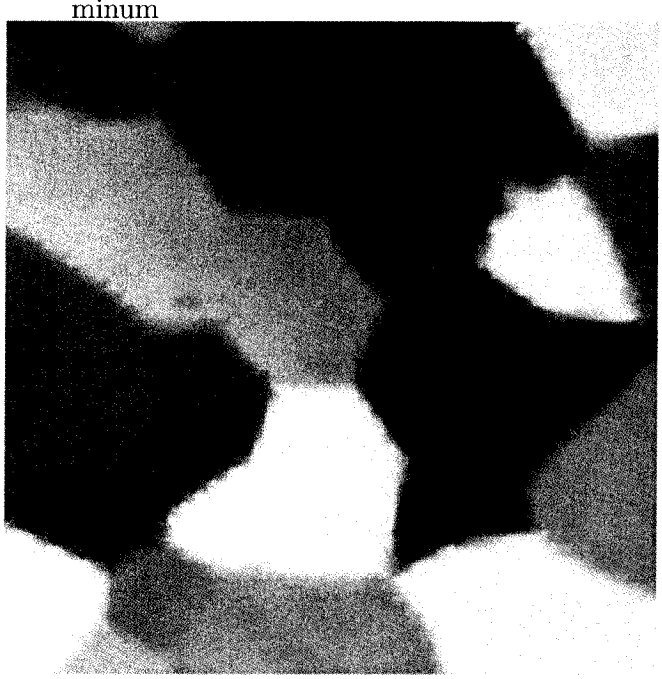
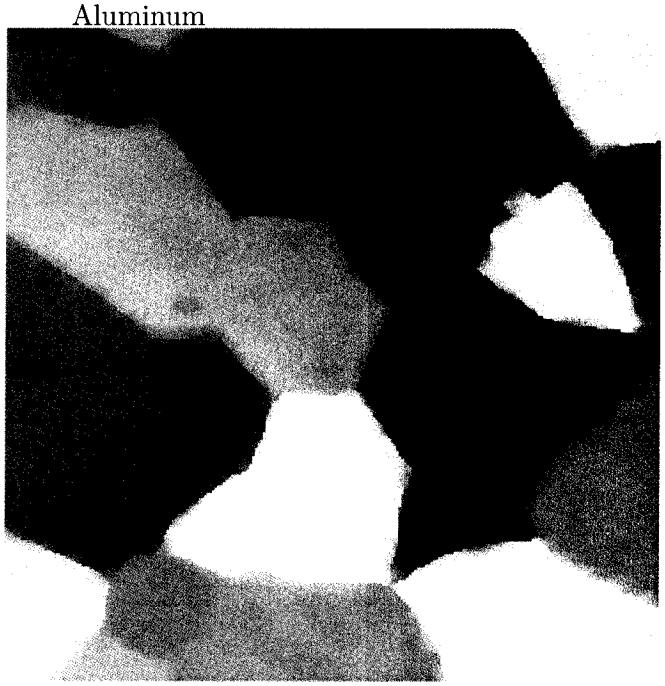


Fig. 5.16 25× Improved Edge-Directed Aluminum



Figures 5.15 and 5.16 show the results obtained from the original and improved edge-directed algorithms, respectively. Observe how the improved algorithm is able to enhance the contrast at edges. Once again, edges have fewer jagged artifacts. Also, note how Figure 5.16 has patches that appear blurred along, but not on, edges. This is due to the improved edge detector's failure at detecting weaker edges that are close to stronger ones.

Fig. 5.17 25× Statistical Aluminum

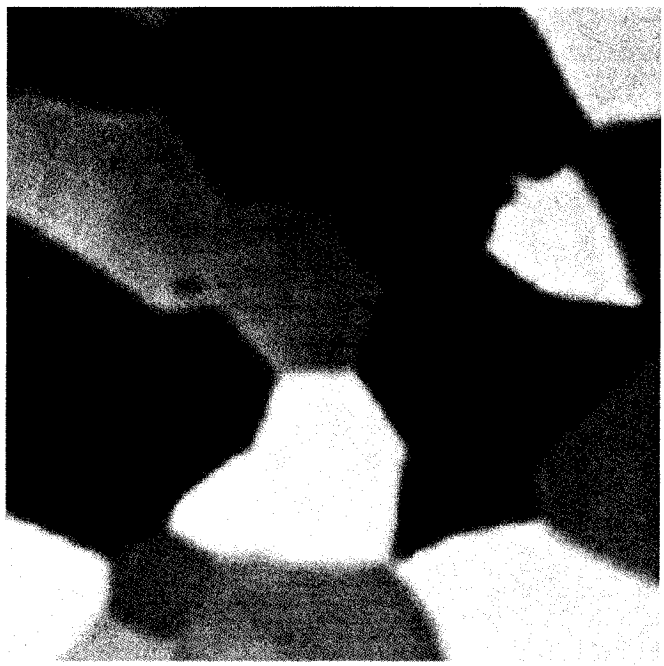
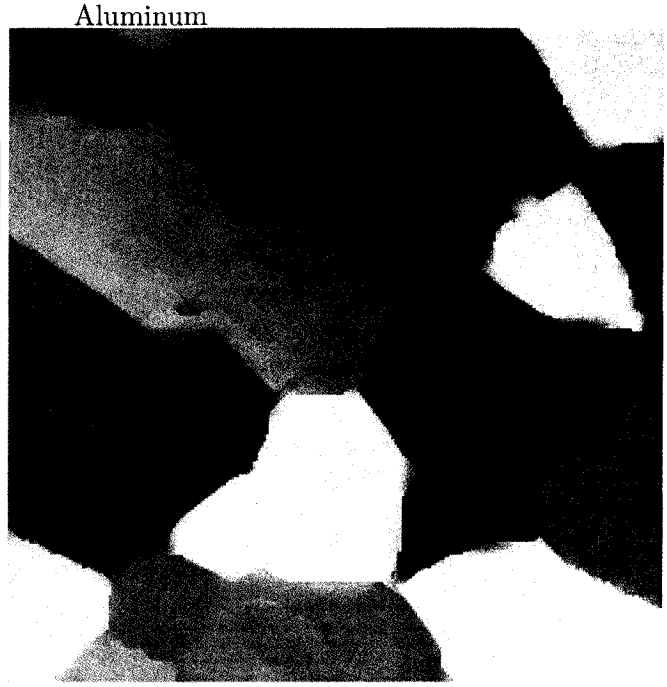


Fig. 5.18 25× Improved Statistical Aluminum

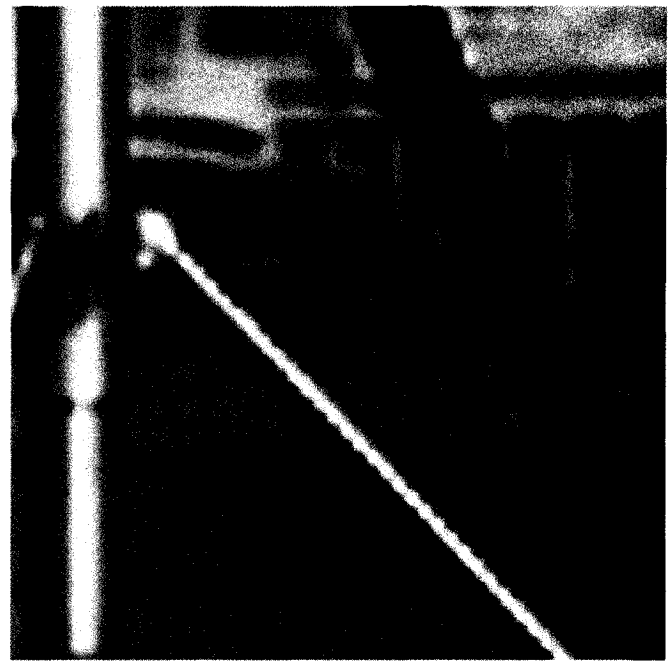


Figures 5.17 and 5.18 show the results obtained from the original and improved statistical algorithms, respectively. Observe the improved algorithm's significant contrast enhancement. Edges are very sharp and clearly delimit the boundaries of the constant regions. Also, due to the nature of this image, the effect of the limited clique directions is not apparent as it was for previous images.

Comparing Figure 5.16 and 5.18 to the bicubic interpolation of Figure 5.14, they both show improvements at the edges with respect to edge sharpness and enhanced contrast. However, the improved statistical algorithm in Figure 5.18 shows better quality than the improved edge-directed in Figure 5.16 by having smoothed-out the blurred patches along edges. This is due to the fact that the improved statistical algorithm is more forgiving of the edge detector's failure to find weaker edges that are close to stronger ones. Instead of improperly interpolating the missed edges, they are smoothed along the edge, which reduces the patches introduced by the improved edge-directed algorithm.

Fig. 5.19 LR Tripod Bar

Fig. 5.20 25× Bicubic Tripod Bar



The final LR test image is shown in Figure 5.19. This image was chosen due to it's awkwardly oriented edges. The term awkward refers to the difficulty with which magnification algorithms have in dealing with oriented edges. Too often it's the case that an algorithm performs well for edges oriented at certain angles, but fails completely for others. The problem is that although an algorithm may perform well at a particular angle, it can fail if the angle is slightly changed. This image is potent due to the relatively close orientation of the three tripod bars. It allows testing an algorithm at 90°, 45° and a midpoint angle near 60°. Figure 5.20 is the bicubic interpolation.

Fig. 5.21 25× Edge-Directed Tripod Bar

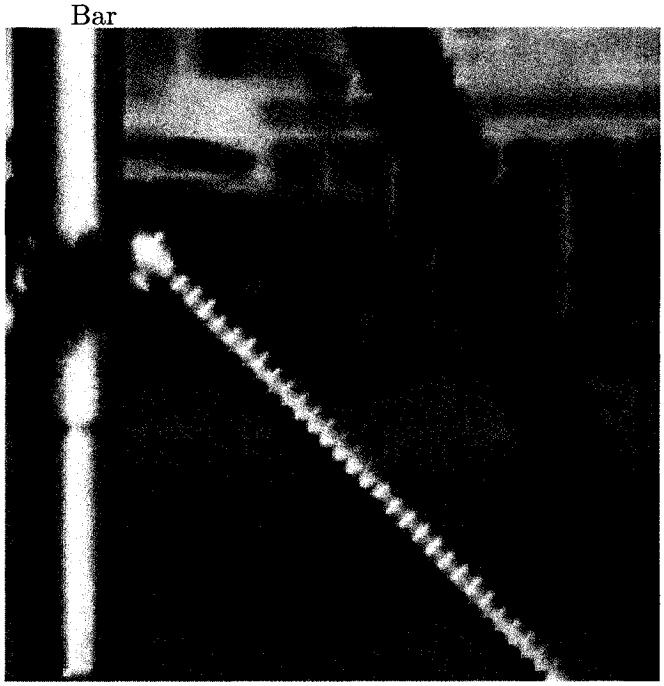
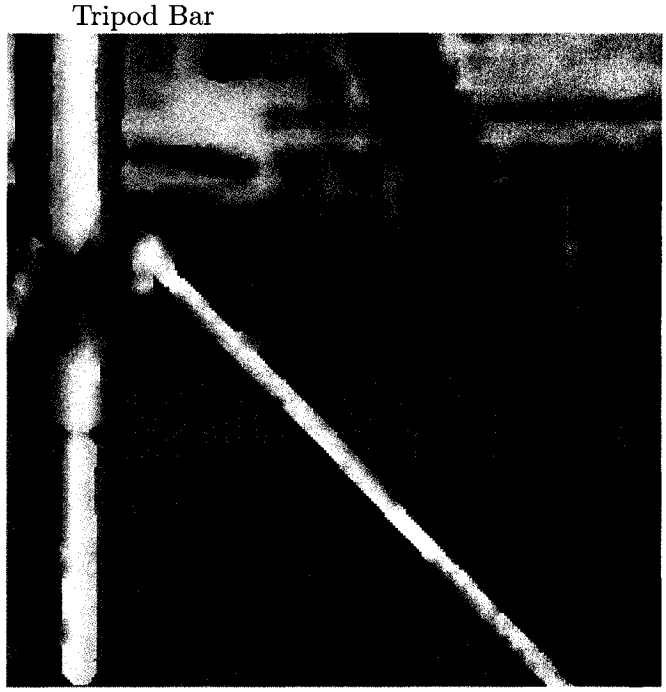


Fig. 5.22 25× Improved Edge-Directed Tripod Bar



Figures 5.21 and 5.22 show the results obtained from the original and improved edge-directed algorithms, respectively. Observe the significant improvement of the narrow-edged bar. In this case, the new edge detection strategy is able to completely eliminate the jagged edges. However, along the more sharply angled, solid black pole, the edges appear to be smeared outwards. The smearing is a result of applying bilinear interpolation to LR unit cells that should have been classified as edges. This failure stems from the previously mentioned deficiencies in the new edge detector.

Fig. 5.23 25× Statistical Tripod Bar

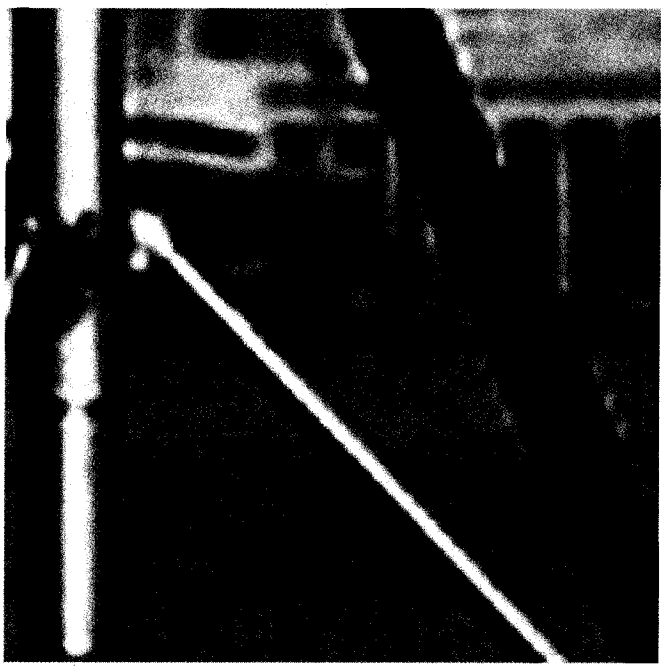
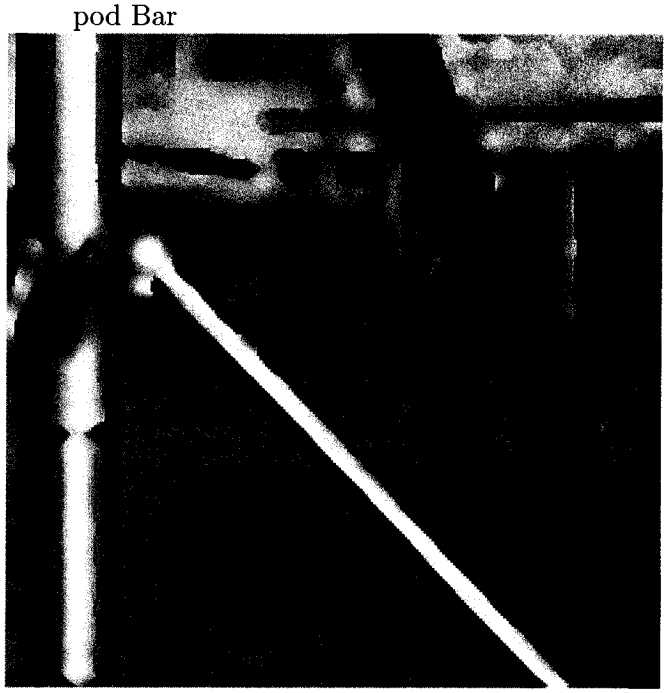


Fig. 5.24 25× Improved Statistical Tripod Bar



Figures 5.23 and 5.24 show the results obtained from the original and improved statistical algorithms, respectively. Observe the improved algorithm's lack of ringing along all edges in the image. Also, due to the directional smoothing, the edges appear straighter and clearly separated from the background.

Unfortunately, the lack of smoothing across edges ($\lambda_2 = 0$) can be seen to introduce incorrect reflections on the vertical support pole of the tripod, located on the left side of the image. Also, note how the grass in the background appears artificial. This image demonstrates the trade off that must be made between forcing sharp edges and allowing for smoothness. The background can be made smoother by increasing λ_2. This will reduce the effect of the edges detected in the background. However, foreground edges will also be smoothed.

Comparing Figure 5.22 and 5.24 to the bicubic interpolation of Figure 5.20, they both show improvements in dealing with various edge orientations. This is seen in the reduced amount of jagged edges. The improved statistical algorithm in Figure 5.24 has sharper

edges and greater contrast than the improved edge-directed algorithm in Figure 5.22. Unfortunately, it suffers from an artificial look to the background. The background of the improved edge-directed algorithm is more consistent with what one would expect it to look like.

From the above test images, it is clear that the proposed improvements made to the edge-directed and statistical algorithms boast an overall better image quality compared to the original algorithms.

Chapter 6

Conclusions

6.1 Summary

This thesis provided a comparison of two adaptive image magnification algorithms selected from the literature. Each algorithm was then improved based on experimental observations from their implementations.

The first selected algorithm, [16], was improved by increasing the accuracy of the edge detection stage. This was accomplished by selecting a more sensitive second-order operator for detecting edges and by introducing a topological estimation technique for determining the path that an edge takes through an LR unit cell. Also, the edge-directed rendering stage was simplified by combining the approaches of [16] and its predecessor, [19]. The new rendering stage applies bilinear interpolation to groups of HR samples within an LR unit cell. The groups are determined by the separations created by the edges passing through the cell. Also, edges are now modelled by a slope-controllable polynomial, giving an additional parameter to control the interpolation.

Results show that the improvements boast an increase in edge sharpness in the interpolated image when compared to [16] and bicubic interpolation. Unfortunately, known limitations of the edge detection stage introduce artifacts when weaker edges are in proximity to stronger ones.

The second selected algorithm, [29], was improved by reducing the amount of smoothing performed across discontinuities. This was accomplished by introducing a set of weights, one for each of the four cliques. The weights are derived from the estimated HR edge map such that each sample in the HR estimate permits or blocks a particular clique based on whether or not that clique crosses a discontinuity. Also, two regularization parameters, now applied to the smoothing term, are used to control the optimization, adding further control over the smoothness of the final estimate.

Results show that the improvements greatly reduce the smoothing in regions containing discontinuities when compared to [29] and bicubic interpolation. Unfortunately, the use of a binary edge map is too rigid to allow smoothing along edges of arbitrary orientation. This is known to cause more smoothing across, and not enough smoothing along, edges whose orientations do not align with the four cliques.

6.2 Thesis Contributions

This thesis has made the following original contributions:

1. A comprehensive survey of the current literature on the topic of adaptive image magnification was performed. From this survey, two promising algorithms were selected.

2. These two algorithms were fully implemented to be as close as possible to the algorithms described in the literature. The algorithms were implemented using the Matlab scripting language.

3. The performance of each algorithm was analyzed by determining the effects of varying the tuning parameters. A summary of the effects was presented.

Following the implementation and analysis, improvements were made to each algorithm.

4. For the edge-directed algorithm, a new edge detection method was developed which achieves the greater sensitivity and accuracy needed for edge-directed interpolation.

Also, a new rendering method was proposed with the goal of simplifying the approach used by the authors of the original algorithm.

5. For the statistical algorithm, a new weighting scheme was introduced to eliminate blurring across edges. The weights are determined from an estimation of the binary edge map of the HR image. Each clique is weighted individually based on proximity to an edge.

6. Following an analysis of the effects of varying the tuning parameters of the improved algorithms, a comparison of the two original and two improved algorithms was performed. The comparison used parameters that were deemed optimal for each respective algorithm. The optimality of the parameters was determined using a subjective visual criterion.

7. All four algorithms, the two originals and their improved versions, were combined into a single environment with a graphical user interface. This interface provides a visual means of entering each algorithm's required parameters, selecting the image to be magnified and specifying the output file in which to save the HR estimate. In addition, the interface provides options for displaying algorithm specific details, such as convergence of the objective function at each iteration, estimated HR binary edge maps, as well as the HR estimate itself. This interface was also implemented in Matlab.

6.3 Further Work

Further work on the edge-directed algorithm would involve finding a means of eliminating the artifacts by improving the accuracy of the edge detection. Essentially, every LR unit cell should be interpolated as if it contained an edge, except the case in which all four LR samples have the same values. This way, weak edges that are near stronger ones will be properly interpolated.

Further work on the statistical algorithm would involve the exploration of alternative approaches for the determination of the clique weights. Instead of deriving the weights from a binary edge map, it may be possible to block or permit smoothing based on a measure of the direction of the gradient at each sample in the LR image. Since the four cliques cover a limited and fixed range of orientations, this approach would allow for smoothing along discontinuities of arbitrary orientation by combining pairs of the fixed cliques. The basic idea is that by properly weighting pairs of cliques with fixed orientations, a new clique is formed whose orientation is derived from the base pair. The concept is similar to how a single vector can be composed by a combination of arbitrarily oriented base vectors. Although the base cliques do not change, the actual cliques used for estimating an HR image are created specifically for that image from the features of the LR image. It is believed that this approach would help to straighten lines and reduce blurring across edges whose orientations are not supported by the four cliques. Another advantage of this approach lies in the reduction of the tuning parameters. Determining the weights directly from the LR image eliminates the need for finding an algorithm's tuning parameters. This would help increase robustness and simplify usage of an algorithm.

An attempt was made at determining the clique weights by estimating the angle of the gradient at each LR sample. The approach separated the 360° cartesian plane into slices of 45° increments (due to the directions of the four cliques). Since the angle of the gradient is perpendicular to an edge, the new clique should be oriented perpendicularly to the gradient. The base cliques used to form the new clique are chosen such that the angle of the new clique falls within the 45° slice formed by the base pair. The weights are the constant coefficients needed to form a unit vector in the direction of the new clique from the unit vectors in the direction of the two base cliques. For any particular edge, only the base cliques have non-zero weights. Once all the weights for the LR samples are known, the weights of the HR samples are found by spreading the weights of each respective clique. For the purposes of conducting initial trials, the weights were spread using a simple zero-order hold filter. This ensures that there will be no overlap between

weights within a particular clique. Although results were promising, the HR estimates suffered from blocking caused by the simplistic method used to spread the weights onto the HR grid. Unfortunately, a more accurate technique was not found.

Another possible improvement of interest lies in the use of different sensor models. In [34], a technique for estimating the sensor model from the LR image is proposed. Preliminary tests found that this approach improved the contrast and reduced ringing in regions containing discontinuities. Applications of the sensor model to blurred images may allow for integrating a measure of restoration into the interpolation algorithm.

Changes in the sensor model also highlight the need for different initial conditions. With a better initial condition, blocking artifacts may be removed more completely and more rapidly.

Appendix A

Detailed Description of the Edge-Directed Algorithm

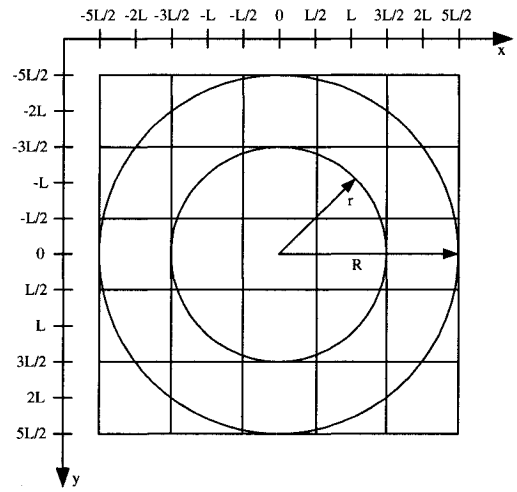
This appendix provides a detailed description of the theory and implementation of the edge-directed magnification algorithm of [16].

A.1 Theory

A.1.1 Edge Detection Filter

The edge detection filter is derived from sampling the continuous-space filter shown in Figure A.1. The sampling is performed on the square grid overlayed onto the continuous filter. The filter has three distinct regions: i) a center region, V_C, of radius r, ii) a surround region, V_S, contained between the radius r and R, and iii) a zero region, V_Z, which includes everything beyond the radius R. The amplitude of the continuous filter in each of the regions is $V_C = A_S$, $V_S = -A_C$ and $V_Z = 0$, where $A_C = \pi r^2$ and $A_S = \pi R^2 - A_C = \pi(R^2 - r^2)$ are the areas of the center and surround regions, respectively. The coefficients of the discrete filter that are located on the boundary between two regions of the continuous filter are computed by averaging the values of both regions weighted by

Fig. A.1 Edge Detection Filter



the portion of the samples area that is contained in each region.

$$\frac{V_{R_1}P_{R_1} + V_{R_2}P_{R_2}}{L^2} = \frac{V_{R_1}P_{R_1} + V_{R_2}(1 - P_{R_1})}{L^2},$$

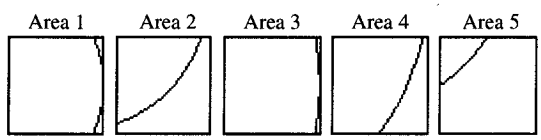
where V_{R_1} and V_{R_2} are the values of two regions of the continuous filter, and P_{R_1} is the portion of a samples area that is contained in region R_1.

From the circular symmetry of the filter, there are only six independent coefficients. The discrete filter can therefore be constructed by

$$h_e[x, y] = \begin{bmatrix} h_5 & h_4 & h_2 & h_4 & h_5 \\ h_4 & h_3 & h_1 & h_3 & h_4 \\ h_2 & h_1 & h_0 & h_1 & h_2 \\ h_4 & h_3 & h_1 & h_3 & h_4 \\ h_5 & h_4 & h_2 & h_4 & h_5 \end{bmatrix}.$$

Of the six coefficients, there are five that are contained in two regions. This results in five distinct area portions that must be computed, as shown in Figure A.2. These area

Fig. A.2 Area Portions of the Independent Filter Coefficients



portions can be computed using,

$$\begin{aligned}
P_{R_1} &= \frac{1}{L^2}\int_{y_0}^{y_1}\int_{x_0}^{\sqrt{R_1^2-y^2}} \mathrm{d}x\mathrm{d}y \\
&= \frac{1}{L^2}\int_{y_0}^{y_1}\left(\sqrt{R_1^2-y^2}-x_0\right)\mathrm{d}y \\
&= \frac{1}{L^2}\left[y\frac{\sqrt{R_1^2-y^2}}{2}+\frac{R_1^2}{2}\arcsin\left(\frac{y}{R_1}\right)-x_0y\right]_{y_0}^{y_1} \\
&= \frac{1}{L^2}\left(y_1\frac{\sqrt{R_1^2-y_1^2}}{2}-y_0\frac{\sqrt{R_1^2-y_0^2}}{2}+\frac{R_1^2}{2}\left[\arcsin\left(\frac{y_1}{R_1}\right)-\arcsin\left(\frac{y_0}{R_1}\right)\right]-x_0(y_1-y_0)\right),
\end{aligned}$$

where R_1 is the radius of the first region.

Table A.1 lists the relevant parameters for each of the area portions in Figure A.2. Note that the area portions are independent of the sample dimension L due to the normalization.

Table A.1 Area Portions

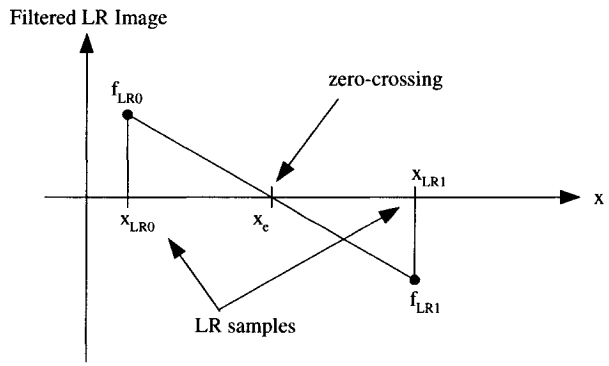
	x_0	y_0	y_1	R_1	P_{R_1}
Area 1	L/2	-L/2	L/2	3L/2	0.9717
Area 2	L/2	L/2	$\sqrt{2}L$	3L/2	0.5454
Area 3	3L/2	-L/2	L/2	5L/2	0.9832
Area 4	3L/2	L/2	3L/2	5L/2	0.7693
Area 5	3L/2	3L/2	2L	5L/2	0.1369

From the area portions in Table A.1, the independent coefficients are,

$$h_0 = 12.5664, h_1 = 12.0115, h_2 = -6.9501, h_3 = 3.6404, h_4 = -5.4380, h_5 = -0.9674.$$

A.1.2 Zero-Crossing Estimation

Fig. A.3 Determination of Zero-Crossings



The line joining the two adjacent LR samples in Figure A.3 is given by

$$f(x) = \left(\frac{f_{LR1} - f_{LR0}}{x_{LR1} - x_{LR0}}\right)(x - x_{LR0}) + f_{LR0}.$$

The zero-crossing, x_e, is found by computing the root of the line.

$$\begin{aligned} f(x_e) &= 0 \\ \left(\tfrac{f_{LR1}-f_{LR0}}{x_{LR1}-x_{LR0}}\right)(x_e - x_{LR0}) + f_{LR0} &= 0 \\ \Rightarrow \qquad x_e &= -\left(\tfrac{x_{LR1}-x_{LR0}}{f_{LR1}-f_{LR0}}\right) f_{LR0} + x_{LR0} \end{aligned}$$

For convenience, it can be assumed that $x_{LR0} = 0$ and $x_{LR1} = 1$. This convention reduces the above to

$$x_e = -\left(\frac{1}{f_{LR1} - f_{LR0}}\right) f_{LR0}.$$

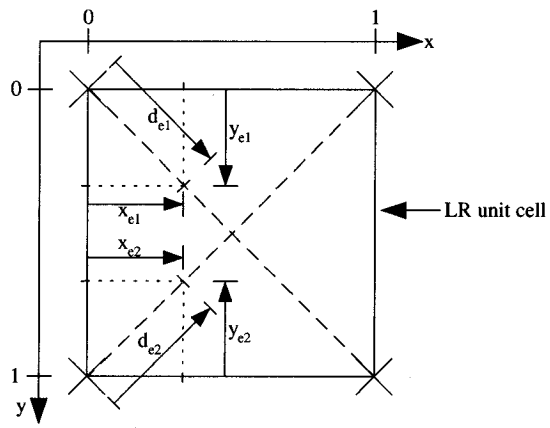
If the zero-crossing is on a diagonal, this convention still holds by noting that $x_{LR1} = \sqrt{2}$. When projecting the diagonal zero-crossing onto the horizontal and vertical axes to obtain

its (x, y) coordinates,

$$\begin{aligned} x &= x_e \cos(45°), \\ y &= x_e \sin(45°), \end{aligned}$$

where $\cos(45°) = \sin(45°) = \frac{1}{\sqrt{2}}$. Therefore, the $\sqrt{2}$ factor is cancelled and the (x, y) coordinates are computed directly instead of passing through the intermediate step.

Fig. A.4 Conventions for Zero-Crossing Computation



Attention must be taken with regards to the convention used for the direction of the axes. Figure A.4 depicts the convention used for computing zero-crossings on the two diagonals. The (x, y) coordinates of the zero-crossings are indicated with respect to the origin used in computing them. Note that this choice of convention requires that the coordinate y_{e2} on the second diagonal be reversed in order to convert it to follow the convention of the global y axis.

A.1.3 Pre-processing

The pre-processing searches for samples from the filtered LR image that have different signs from their eight-nearest neighbors. If such a sample is found, it is replaced by the

mean of its eight neighbors. If the mean is zero, then the median is used.

A.1.4 Rendering

Rendering is performed based on the type of edge passing through an LR unit cell. For LR unit cells without edges, bilinear interpolation is applied by using,

$$\begin{aligned} HR[x,y] &= \left(\frac{1-x}{Mx}\right)\left(\frac{1-y}{My}\right)c_1 \\ &+ \left(\frac{1-x}{Mx}\right)\left(\frac{y}{My}\right)c_2 \\ &+ \left(\frac{x}{Mx}\right)\left(\frac{y}{My}\right)c_3 \\ &+ \left(\frac{x}{Mx}\right)\left(\frac{1-y}{My}\right)c_4, \end{aligned}$$

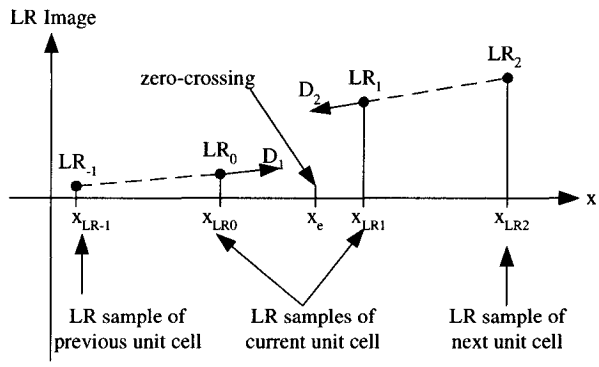
where M_x and M_y are the horizontal and vertical magnification factors, and the c_i are the corner samples of the LR unit cell taken in the anti-clockwise direction, starting from the top-left corner (as shown in Figure A.6).

For LR unit cells with edges, the rendering is edge-directed. The rendering is further separated by the number of zero-crossings on the borders of each LR unit cell. To begin, the case of two zero-crossings will be treated. Four zero-crossings is simply an extension of the method for two. No other number of zero-crossings is possible.

The first step is in specifying the polynomial curve used to model the edges. Figure A.5 shows the relevant parameters used for fitting a curve to an edge. The edge curve is defined as a piece-wise continuous polynomial composed of two segments. The two segments are given by

$$\begin{aligned} f_1(t) &= \Delta_1[\hat{D}_1 t + (1-\hat{D}_1)t^n] + LR_0, \quad t = \tfrac{x-x_{LR0}}{x_e-x_{LR0}} \quad for \quad x \in [x_{LR0}, x_e], \\ f_2(s) &= \Delta_2[\hat{D}_2 s + (1-\hat{D}_2)s^n] + LR_1, \quad s = \tfrac{x-x_{LR1}}{x_e-x_{LR1}} \quad for \quad x \in [x_e, x_{LR1}], \end{aligned}$$

Fig. A.5 Edge Curve Parameters



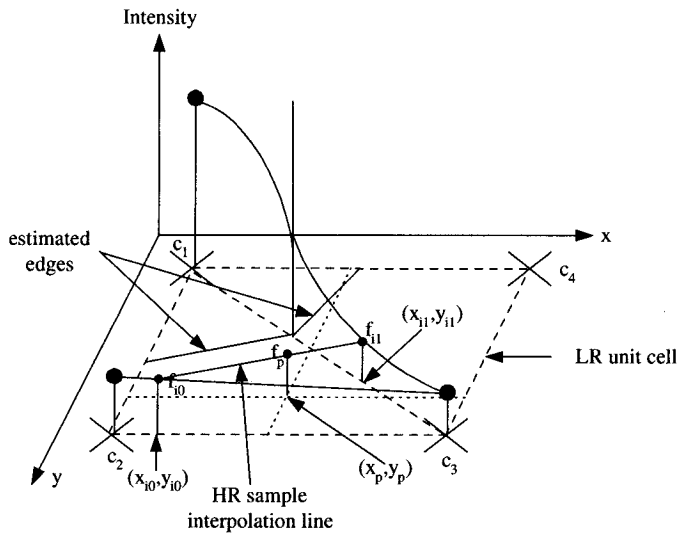
with

$$
\begin{aligned}
D_1 &= LR_0 - LR_{-1}, \\
D_2 &= LR_1 - LR_2, \\
\Delta_E &= LR_1 - LR_0, \\
k &= \frac{x_e - x_{LR0}}{x_{LR1} - x_e}, \\
n &= \lfloor 1 + (x_{LR1} - x_{LR0} - 1)\tanh(\alpha\Delta_E) \rfloor, \\
\Delta_2 &= \frac{(n-1)(D_1 + kD_2)}{(1+k)n} - \frac{\Delta_E}{1+k}, \\
\Delta_1 &= \Delta_2 + \Delta_E, \\
\hat{D}_1 &= \frac{D_1}{\Delta_1}, \\
\hat{D}_2 &= \frac{D_2}{\Delta_2}.
\end{aligned}
$$

The edge curve is used directly to interpolate the HR samples along cell borders and/or diagonals with edges. Borders that do not have edges are interpolated linearly.

For HR samples contained within the LR unit cell, the scheme shown in Figure A.6 is used. This figure shows a cell of a top-left corner edge type. The edge along the diagonal is fitted with the curve, and the border without an edge is fitted with a straight line. The

Fig. A.6 HR Sample Interpolation



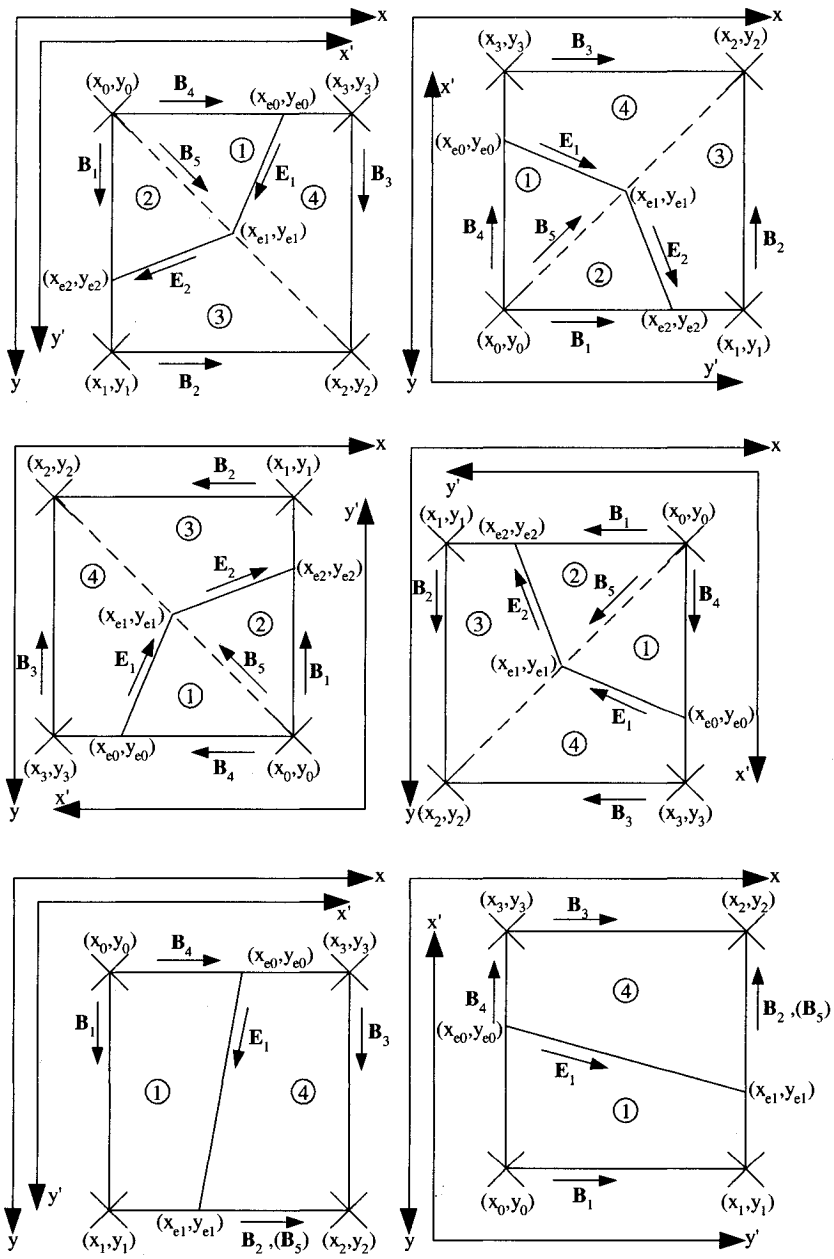
HR sample at coordinates (x_p, y_p) is interpolated linearly along a line parallel to the edge and passing through the sample. The end-points of the line are obtained by evaluating the fitted curve and line at the intersections with the two boundaries. The interpolation of the HR sample is therefore,

$$HR[x_p, y_p] = \frac{f_{i1} - f_{i0}}{\sqrt{(x_{i1} - x_{i0})^2 + (y_{i1} - y_{i0})^2}} \sqrt{(x_p - x_{i0})^2 + (y_p - y_{i0})^2} + f_{i0},$$

where f_{i0} and f_{i1} are the values at the intersecting end-points (x_{i0}, y_{i0}) and (x_{i1}, y_{i1}), respectively.

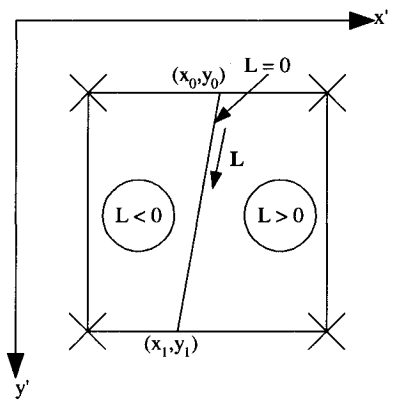
It can be observed that the interpolation scheme for the example in Figure A.6 can be generalized for all edge types with two zero-crossings on the borders of the LR unit cell. The primary difference between the edge types with two zero-crossings is in the location of the LR corner sample that is separated by the edge. Figure A.7 shows the six edge types with two zero-crossings on the border. The global coordinate system is indicated by the x and y axes, whereas the coordinate system relative to the edge separated LR corner

Fig. A.7 Edge-Directed Rendering Parameters for Two Zero-Crossings



sample is indicated by the x' and y' axes. The coordinates of the four LR corner samples are given with respect to x' and y'. The origin, (x_0, y_0), is always assigned to the edge-separated LR corner sample. For each figure, the boundaries of the LR unit cell are given by the vectors $\mathbf{B}_i$, whose directions are indicated by the arrows. The estimated edges are also assigned vectors $\mathbf{E}_i$ and directions. The circled numbers indicate the different interpolation regions within the cell. Note that the boundaries of the cell are also regions. For simplicity, they were not labelled in the figure, however they will be treated in the following. For the two last edge types in the figure, the diagonal boundary is meaningless. In order to generalize the interpolation, the vector used on the diagonal boundary, $\mathbf{B}_5$, is set equal to the boundary vector $\mathbf{B}_2$ on the border of the cell.

Fig. A.8 Detecting the Different Interpolation Regions Within an LR Unit Cell



The four regions within an LR unit cell can be determined by applying thresholds to the linear equations for each of the vectors in Figure A.7. The idea is illustrated in Figure A.8, where three thresholds are applied to the line equation of the vector **L**. The x' and y' axes used in Figure A.8 shows that the same principle applies regardless of the relative origin used by the coordinate system.

The line equation of a vector **L** defined by the end-points (x_0, y_0) and (x_1, y_1), such

that **L** is oriented from (x_0, y_0) to (x_1, y_1), is

$$L(x, y) = a(x - x_0) + b(y - y_0).$$

For this line,

$$\begin{aligned} a(x_1 - x_0) + b(y_1 - y_0) &= 0 \\ \frac{a}{b} &= -\frac{y_1 - y_0}{x_1 - x_0}. \end{aligned}$$

Choosing $a = y_1 - y_0$ and $b = x_0 - x_1$ results in, $L(x, y) = (y_1 - y_0)(x - x_0) + (x_0 - x_1)(y - y_0)$. Figure A.8 shows the three regions obtained by applying the thresholds $L = 0$, $L < 0$ and $L > 0$. A region in Figure A.7 can be determined by combinations of the relevant thresholds applied to the line equation of each vector that bounds the region. Table A.2 shows the equations for determining the regions. The thresholds are applied to all HR samples within, and on the boundaries of, an LR unit cell. The operator '$\cdot$' denotes logical AND. Note that regions 5 through 9 are actually boundaries of the cell.

Table A.2 Region Detection Equations For Two Zero-Crossings

	Region Equation	Bounding Lines
Region 1	$(E_1 \leq 0) \cdot (B_1 > 0) \cdot (B_4 < 0) \cdot (B_5 > 0)$	B_4 and B_5
Region 2	$(E_2 \leq 0) \cdot (B_1 > 0) \cdot (B_5 < 0)$	B_1 and B_5
Region 3	$(E_2 > 0) \cdot (B_1 > 0) \cdot (B_2 > 0) \cdot (B_5 < 0)$	(B_1 and B_5) or (B_2 and B_5)
Region 4	$(E_1 > 0) \cdot (B_4 < 0) \cdot (B_3 < 0) \cdot (B_5 > 0)$	(B_4 and B_5) or (B_3 and B_5)
Region 5 (Line B_1)	$(B_1 = 0) \cdot (B_2 > 0)$	B_1
Region 6 (Line B_2)	$(B_2 = 0) \cdot (B_3 < 0)$	B_2
Region 7 (Line B_3)	$(B_3 = 0) \cdot (B_4 < 0)$	B_3
Region 8 (Line B_4)	$(B_4 = 0) \cdot (B_1 > 0)$	B_4
Region 9 (Line B_5)	$(B_5 = 0) \cdot (B_2 > 0) \cdot (B_4 < 0)$	B_5

Once the region of a particular HR sample is known, it can be interpolated according to its bounding lines. The bounding lines for each region are given Table A.2. Knowledge

of the bounding lines allows for the determination of the intersecting end-points of the line parallel to the edge in that region. The intersections (x_{i0}, y_{i0}) and (x_{i1}, y_{i1}), shown in Figure A.6, are found by

$$\begin{aligned} L_1 : \\ L_2 : \end{aligned} \begin{bmatrix} a_1 & b_1 \\ a_2 & b_2 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} - \begin{bmatrix} a_1 x_{01} + b_1 y_{01} \\ a_2 x_{02} + b_2 y_{02} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} x_i \\ y_i \end{bmatrix} = \begin{bmatrix} a_1 & b_1 \\ a_2 & b_2 \end{bmatrix}^{-1} \begin{bmatrix} a_1 x_{01} + b_1 y_{01} \\ a_2 x_{02} + b_2 y_{02} \end{bmatrix}.$$

where L_1 is a bounding line and L_2 is the line parallel to the edge.

For regions 3 and 4, there are two mutually exclusive sets of bounding lines. In both cases, the intersections with the first set are computed. If either the x or y coordinate of one of the intersecting end-points is outside of the LR unit cell (i.e outside the range $[x_{LR0}, x_{LR1}]$ and $[y_{LR0}, y_{LR1}]$), then that intersection must be re-computed using the second set of bounding lines.

Fig. A.9 Edge-Directed Rendering Parameters for Four Zero-Crossings

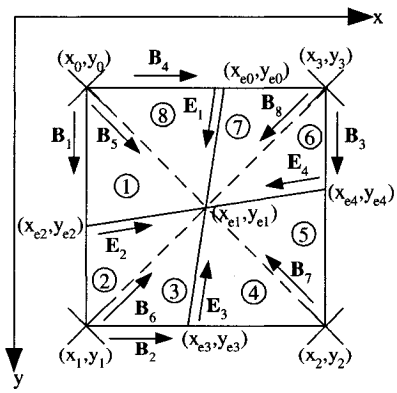


Figure A.9 shows the interpolation regions in the case of zero-crossings on all four borders of an LR unit cell. Determination of the different regions is given by Table

Table A.3 Region Detection Equations For Four Zero-Crossings

	Region Equation	Bounding Lines
Region 1	$(E_2 \geq 0) \cdot (B_1 > 0) \cdot (B_5 < 0)$	B_1 and B_5
Region 2	$(E_2 < 0) \cdot (B_1 > 0) \cdot (B_6 \geq 0)$	B_1 and B_6
Region 3	$(E_3 \geq 0) \cdot (B_2 > 0) \cdot (B_6 < 0)$	B_2 and B_6
Region 4	$(E_3 < 0) \cdot (B_2 > 0) \cdot (B_7 \geq 0)$	B_2 and B_7
Region 5	$(E_4 \geq 0) \cdot (B_3 < 0) \cdot (B_7 < 0)$	B_3 and B_7
Region 6	$(E_4 < 0) \cdot (B_3 < 0) \cdot (B_8 \geq 0)$	B_3 and B_8
Region 7	$(E_1 \geq 0) \cdot (B_4 < 0) \cdot (B_8 < 0)$	B_4 and B_8
Region 8	$(E_1 < 0) \cdot (B_4 < 0) \cdot (B_5 \geq 0)$	B_4 and B_5
Region 9 (Line B_1)	$(B_1 = 0) \cdot (B_2 > 0)$	B_1
Region 10 (Line B_1)	$(B_2 = 0) \cdot (B_3 < 0)$	B_2
Region 11 (Line B_1)	$(B_3 = 0) \cdot (B_4 < 0)$	B_3
Region 12 (Line B_1)	$(B_4 = 0) \cdot (B_1 > 0)$	B_4

A.3. Note that each region only has a single set of bounding lines. This implies that the intersections are always contained within the LR unit cell. Interpolation within each region is the same as in the case of two zero-crossings on the borders.

A.1.5 Correction

The correction stage successively approximates the HR estimate by means of a feedback loop. The loop applies a correction to the LR image input to the edge-directed interpolation stage such that a decimated version of the HR estimate at the current iteration is forced to match the original LR image. The feedback loop is characterized by

$$\begin{aligned} HR_0^{(it)}[x,y] &= h_0[x,y] * HR^{(it)}[x,y], \\ \downarrow HR_0^{(it)}[x,y] &= HR_0^{(it)}[xM_x, yM_y], x = 0,1,..X_{LR}-1, y = 0,1,..Y_{LR}-1, \\ LR^{(it+1)}[x,y] &= LR^{(it)}[x,y] - \alpha\left(\downarrow HR_0^{(it)}[x,y] - LR[x,y]\right), \end{aligned}$$

where h_0 is the moving average filter with M_y rows and M_x columns, and α is the loop gain. The initial condition $LR^{(0)}[x,y]$ is simply the original input image $LR[x,y]$.

A.2 Implementation

The edge-directed algorithm can be implemented by the following steps:

1. Filter the LR image with the edge detection filter,

$$LR_e[x, y] = h_e[x, y] * LR[x, y].$$

2. Apply the pre-processing strategy to $LR_e[x, y]$.

3. Compute the zero-crossings in the horizontal and vertical directions between all pairs of samples of different sign in $LR_e[x, y]$.

4. Classify each LR unit cell according to the number of zero-crossings on the four borders of the cell (using the edge types of Figure 3.5 from Section 3.1) of Chapter 3. If a cell is classified as a corner edge, compute the zero-crossing along the appropriate diagonal.

5. Determine the dimensions of the HR estimate using

$$\begin{aligned} X_{HR} &= (M_x - 1)(X_{LR} - 1) + X_{LR}, \\ Y_{HR} &= (M_y - 1)(Y_{LR} - 1) + Y_{LR}, \end{aligned}$$

   where M_x and M_y are the horizontal and vertical magnification factors, and X_{LR} and Y_{LR} are the horizontal and vertical dimensions of the LR image.

6. For each iteration *it* of the algorithm:

   (a) Apply edge-directed rendering to each LR unit cell containing an edge. If a cell has no edges, apply bilinear interpolation.

   (b) Apply the correction to the LR input.

Appendix B

Detailed Description of the Improved Edge-Directed Algorithm

This appendix provides a detailed description of the theory and implementation of the improved edge-directed magnification algorithm proposed in Section 3.2 of Chapter 3.

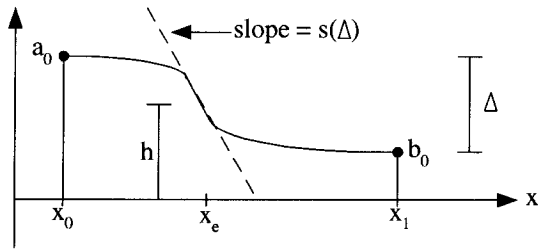
B.1 Theory

B.1.1 Edge Detection Filter

The edge detection filter is derived from sampling the circularly symmetric, continuous-space Laplacian-of-Gaussian filter. The discrete coefficients are computed as in [36] using,

$$\begin{aligned} h_g[x,y] &= \exp\left(-\frac{x^2+y^2}{2\sigma^2}\right), \\ h_e[x,y] &= \frac{x^2+y^2-2\sigma^2}{2\pi\sigma^6 \sum_n \sum_m h_g[m,n]} h_g[x,y]. \end{aligned}$$

Fig. B.1 New Edge Curve Parameters



B.1.2 Rendering

The new edge model described in Section 3.2.2, is specified by

$$p(x) = \begin{cases} p_1(x) \\ p_2(x) \end{cases} = \begin{cases} a_m \left(\frac{x-x_0}{x_e-x_0}\right)^m + a_0, & x_0 \le x \le x_e \\ b_n \left(\frac{x_1-x}{x_1-x_e}\right)^n + b_0, & x_e \le x \le x_1 \end{cases}.$$

Figure B.1 illustrates the variable slope edge model. The free parameters are

$$\begin{array}{lcl} p_1(x_0) = a_0 & & \\ p_1(x_e) = a_m + a_0 = h & \Rightarrow & a_m + a_0 = \frac{a_0+b_0}{2} \\ p_1'(x_e) = s(\Delta) & \Rightarrow & m a_m \frac{1}{x_e-x_0} = s(\Delta) \\ & \Rightarrow & a_m = \frac{b_0-a_0}{2} \\ & \Rightarrow & m = \frac{(x_e-x_0)s(\Delta)}{a_m} \\ p_2(x_1) = b_0 & & \\ p_2(x_e) = b_n + b_0 = h & \Rightarrow & b_n + b_0 = \frac{a_0+b_0}{2} \\ p_2'(x_e) = s(\Delta) & \Rightarrow & -n b_n \frac{1}{x_1-x_e} = s(\Delta) \\ & \Rightarrow & b_n = \frac{a_0-b_0}{2} \\ & \Rightarrow & n = -\frac{(x_1-x_e)s(\Delta)}{b_n} \end{array}$$

The slope control function $s(\Delta)$ is specified as

$$\begin{aligned} s(\Delta) &= A\Delta^3 + B\Delta^2 + C\Delta + D, \\ s'(\Delta) &= 3A\Delta^2 + 2B\Delta + C, \end{aligned}$$

for $0 \leq \Delta \leq 1$, and $-s(-\Delta)$ for $-1 \leq \Delta \leq 0$. The free parameters are

$$\begin{array}{rcl} s(0) = 0 & \Rightarrow & D = 0 \\ s(1) = S & \Rightarrow & A+B+C+D = S \Rightarrow A+B = S \\ s'(0) = 0 & \Rightarrow & C = 0 \\ s'(1) = 0 & \Rightarrow & 3A+2B+C = 0 \Rightarrow 3A+2B = 0 \\ & & \Rightarrow A = -2S \\ & & \Rightarrow B = 3S \end{array}$$

which results in

$$\begin{aligned} s(\Delta) &= -2S\Delta^3 + 3S\Delta^2, \\ &= S\Delta^2(3 - 2\Delta), \end{aligned}$$

where the parameter S is a free parameter of the magnification algorithm.

B.2 Implementation

The improved edge-directed algorithm can be implemented by the following steps:

1. Filter the LR image with the edge detection filter,

$$LR_e[x, y] = h_e[x, y] * LR[x, y].$$

2. Compute the zero-crossings in the horizontal and vertical directions between all pairs of samples of different sign in $LR_e[x, y]$.

3. Classify each LR unit cell as described in Section 3.2.1 of Chapter 3.

4. Determine the dimensions of the HR estimate using

$$\begin{aligned} X_{HR} &= (M_x - 1)(X_{LR} - 1) + X_{LR}, \\ Y_{HR} &= (M_y - 1)(Y_{LR} - 1) + Y_{LR}, \end{aligned}$$

where M_x and M_y are the horizontal and vertical magnification factors, and X_{LR} and Y_{LR} are the horizontal and vertical dimensions of the LR image.

5. For each iteration it of the algorithm:

   (a) Apply the edge-directed rendering scheme described in Section 3.2.2 of Chapter 3.

   (b) Apply the correction to the LR input,

$$\begin{aligned} HR_0[x, y] &= h_0[x, y] * HR[x, y], \\ \downarrow HR_0[x, y] &= HR_0[xM_x, yM_y], x = 0, 1, ..X_{LR} - 1, y = 0, 1, ..Y_{LR} - 1, \\ LR^{(it+1)}[x, y] &= LR^{(it)}[x, y] + \alpha\left(\downarrow HR_0[x, y] - LR[x, y]\right), \end{aligned}$$

where h_0 is the moving average filter with M_y rows and M_x columns, and α is the loop gain.

Appendix C

Detailed Description of the Statistical Algorithm

This appendix provides a detailed description of the theory and implementation of the statistical magnification algorithm of [29].

C.1 Theory

C.1.1 Minimization Problem

The Bayesian estimation problem reduces to the following minimization,

$$\mathbf{HR}_{\min} = \arg\min_{\mathbf{HR}}\{M[\mathbf{HR}, T, \lambda]\}.$$

The objective function is,

$$\begin{aligned} M[\mathbf{HR}, T, \lambda] &= \Omega[\mathbf{HR}, T] + \lambda \|\mathbf{LR} - \mathbf{D}(\mathbf{HR})\|^2 \\ &= \sum_{x=0}^{X_{HR}-1} \sum_{y=0}^{Y_{HR}-1} \sum_{c=0}^{3} \rho_T(\mathbf{d}_{x,y,c}^t \mathbf{HR}) \\ &\quad + \lambda \sum_{x=0}^{X_{LR}-1} \sum_{y=0}^{Y_{LR}-1} \left(y[x,y] - \frac{1}{M_x M_y} \sum_{m=xM_x}^{(x+1)M_x-1} \sum_{n=yM_y}^{(y+1)M_y-1} HR[m,n] \right)^2 \end{aligned}$$

where **LR** is a lexicographically ordered $X_{LR}Y_{LR} \times 1$ vector of samples from the original low-resolution image, **HR** is the $X_{HR}Y_{HR} \times 1$ vector of samples from the high-resolution estimate, **D** is a $X_{LR}Y_{LR} \times X_{HR}Y_{HR}$ constant block-diagonal decimation matrix and

$$\rho_T(x) = \begin{cases} x^2, & |x| \leq T \\ T^2 + 2T(|x| - T), & |x| > T \end{cases}.$$

Note that the dimensions of the HR estimate are such that $X_{HR} = (M_x - 1)(X_{LR} - 1) + X_{LR}$ and $Y_{HR} = (M_y - 1)(Y_{LR} - 1) + Y_{LR}$, where M_x and M_y are the magnification factors in the horizontal and vertical directions, and X_{LR} and Y_{LR} are the horizontal and vertical dimensions of the LR image.

The cliques are defined on second order neighborhoods with the following potentials:

$$\begin{aligned} 1)\ \mathbf{d}_{x,y,0}^t \mathbf{HR} &= HR[x-1,y] - 2HR[x,y] + HR[x+1,y], \\ 2)\ \mathbf{d}_{x,y,1}^t \mathbf{HR} &= \frac{1}{2} HR[x-1,y+1] - HR[x,y] + \frac{1}{2} HR[x+1,y-1], \\ 3)\ \mathbf{d}_{x,y,2}^t \mathbf{HR} &= HR[x,y-1] - 2HR[x,y] + HR[x,y+1], \\ 4)\ \mathbf{d}_{x,y,3}^t \mathbf{HR} &= \frac{1}{2} HR[x-1,y-1] - HR[x,y] + \frac{1}{2} HR[x+1,y+1], \end{aligned}$$

where $HR[x,y]$ is the value of the sample at coordinates (x,y). Note that the evaluation of the clique smoothness measures are equivalent to preforming 2D filtering operations.

The convolution kernel for each clique is therefore,

$$d_0 = \begin{bmatrix} 0 & 0 & 0 \\ 1 & -2 & 1 \\ 0 & 0 & 0 \end{bmatrix}, d_1 = \begin{bmatrix} 0 & 0 & 1/2 \\ 0 & -1 & 0 \\ 1/2 & 0 & 0 \end{bmatrix}, d_2 = \begin{bmatrix} 0 & 1 & 0 \\ 0 & -2 & 0 \\ 0 & 1 & 0 \end{bmatrix}, d_3 = \begin{bmatrix} 1/2 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1/2 \end{bmatrix}.$$

C.1.2 Gradient of the Objective Function

The first step in the optimization procedure is to compute the gradient of the objective function $M[\mathbf{HR}, T, \lambda]$ with respect to every sample in the high-resolution estimate. Let this gradient be defined as,

$$\begin{aligned} \mathbf{g} &= \nabla M[\mathbf{HR}, T, \lambda] \\ &= \nabla \Omega[\mathbf{HR}, T] + \lambda \nabla \|\mathbf{LR} - \mathbf{D}(\mathbf{HR})\|^2 \\ &= \mathbf{g}_1 + \mathbf{g}_2 \end{aligned}$$

where,

$$\nabla = \sum_{x=0}^{X_{HR}-1} \sum_{y=0}^{Y_{HR}-1} \hat{\mathbf{HR}}_{x,y} \frac{\partial}{\partial HR_{x,y}},$$

$\hat{\mathbf{HR}}_{x,y}$ is the unit vector which is one at sample $HR[x, y]$ and zero everywhere else, $\frac{\partial}{\partial HR_{x,y}}$ is the partial derivative taken with respect to sample $HR[x, y]$, $\mathbf{g}_1$ is the gradient of the smoothing term and $\mathbf{g}_2$ is the gradient of the data fidelity term.

C.1.3 Gradient of the Smoothing Term

The gradient of the smoothing term is given by

$$\begin{aligned} \mathbf{g}_1 &= \nabla \Omega[\mathbf{HR}, T] \\ &= \sum_{x=0}^{X_{HR}-1} \sum_{y=0}^{Y_{HR}-1} \hat{\mathbf{HR}}_{x,y} \frac{\partial}{\partial HR_{x,y}} \Omega[\mathbf{HR}, T]. \end{aligned}$$

After expanding the summations in $\Omega[\mathbf{HR}, T]$, the terms containing $HR[x, y]$ are:

$$
\begin{aligned}
1)\quad & \mathbf{d}^t_{x,y,0}\mathbf{HR} = HR[x-1,y] - 2HR[x,y] + HR[x+1,y], \\
2)\quad & \mathbf{d}^t_{x,y,1}\mathbf{HR} = \tfrac{1}{2}HR[x-1,y+1] - HR[x,y] + \tfrac{1}{2}HR[x+1,y-1], \\
3)\quad & \mathbf{d}^t_{x,y,2}\mathbf{HR} = HR[x,y-1] - 2HR[x,y] + HR[x,y+1], \\
4)\quad & \mathbf{d}^t_{x,y,3}\mathbf{HR} = \tfrac{1}{2}HR[x-1,y-1] - HR[x,y] + \tfrac{1}{2}HR[x+1,y+1], \\
5)\quad & \mathbf{d}^t_{x-1,y-1,3}\mathbf{HR} = \tfrac{1}{2}HR[x-2,y-2] - HR[x-1,y-1] + \tfrac{1}{2}HR[x,y], \\
6)\quad & \mathbf{d}^t_{x,y-1,2}\mathbf{HR} = HR[x,y-2] - 2HR[x,y-1] + HR[x,y], \\
7)\quad & \mathbf{d}^t_{x+1,y-1,1}\mathbf{HR} = \tfrac{1}{2}HR[x,y] - HR[x+1,y-1] + \tfrac{1}{2}HR[x+2,y-2], \\
8)\quad & \mathbf{d}^t_{x-1,y,0}\mathbf{HR} = HR[x-2,y] - 2HR[x-1,y] + HR[x,y], \\
9)\quad & \mathbf{d}^t_{x+1,y,0}\mathbf{HR} = HR[x,y] - 2HR[x+1,y] + HR[x+2,y], \\
10)\quad & \mathbf{d}^t_{x-1,y+1,1}\mathbf{HR} = \tfrac{1}{2}HR[x-2,y+2] - HR[x-1,y+1] + \tfrac{1}{2}HR[x,y], \\
11)\quad & \mathbf{d}^t_{x,y+1,2}\mathbf{HR} = HR[x,y] - 2HR[x,y+1] + HR[x,y+2], \\
12)\quad & \mathbf{d}^t_{x+1,y+1,3}\mathbf{HR} = \tfrac{1}{2}HR[x,y] - HR[x+1,y+1] + \tfrac{1}{2}HR[x+2,y+2].
\end{aligned}
$$

Therefore, the first-order partial derivative of $\Omega[\mathbf{HR}, T]$ with respect to each high-resolution sample is,

$$
\begin{aligned}
\frac{\partial}{\partial HR_{x,y}}\Omega[\mathbf{HR}, T] &= \frac{\partial}{\partial HR_{x,y}} \sum_{m=-1}^{1} \Big[\rho_T(\mathbf{d}^t_{(x+m),y,0}\mathbf{HR}) \\
&\quad + \rho_T(\mathbf{d}^t_{(x-m),(y+m),1}\mathbf{HR}) \\
&\quad + \rho_T(\mathbf{d}^t_{x,(y+m),2}\mathbf{HR}) \\
&\quad + \rho_T(\mathbf{d}^t_{(x+m),(y+m),3}\mathbf{HR}) \Big] \\
&= \sum_{m=-1}^{1} \Big[d_0[x+m,y]\rho'_T(\mathbf{d}^t_{(x+m),y,0}\mathbf{HR}) \\
&\quad + d_1[x-m,y+m]\rho'_T(\mathbf{d}^t_{(x-m),(y+m),1}\mathbf{HR}) \\
&\quad + d_2[x,y+m]\rho'_T(\mathbf{d}^t_{x,(y+m),2}\mathbf{HR}) \\
&\quad + d_3[x+m,y+m]\rho'_T(\mathbf{d}^t_{(x+m),(y+m),3}\mathbf{HR}) \Big]
\end{aligned}
$$

where,

$$\rho'_T(x) = \begin{cases} -2T, & x < -T \\ 2x, & |x| \leq T \\ 2T, & x > T \end{cases}.$$

From the above, the contribution of each clique to the gradient is simply the HR estimate filtered by each of the clique smoothness measures, followed by the application of the first-order derivative of the Huber function, followed by another filtering by the clique smoothness measures.

C.1.4 Gradient of the Data Fidelity Term

The gradient of the data fidelity term is given by

$$\begin{aligned} \mathbf{g}_2 &= \lambda \nabla \|\mathbf{LR} - \mathbf{D}(\mathbf{HR})\|^2 \\ &= \lambda \sum_{x=0}^{X_{HR}-1} \sum_{y=0}^{Y_{HR}-1} \hat{\mathbf{HR}}_{x,y} \frac{\partial}{\partial HR_{x,y}} \|\mathbf{LR} - \mathbf{D}(\mathbf{HR})\|^2. \end{aligned}$$

After expanding $\|\mathbf{LR} - \mathbf{D}(\mathbf{HR})\|^2$, there is only one term containing $HR[x, y]$. Therefore, the first-order partial derivative of $\|\mathbf{LR} - \mathbf{D}(\mathbf{HR})\|^2$ with respect to each high-resolution sample is,

$$\frac{\partial}{\partial HR_{x,y}} \|\mathbf{LR}-\mathbf{D}(\mathbf{HR})\|^2 = 2 \left(LR[i,j] - \frac{1}{M_x M_y} \sum_{m=iM_x}^{(i+1)M_x-1} \sum_{n=jM_y}^{(j+1)M_y-1} HR[m,n] \right) \left(-\frac{1}{M_x M_y} \right),$$

where $i = \left\lfloor \frac{x}{M_x} \right\rfloor$ and $j = \left\lfloor \frac{y}{M_y} \right\rfloor$.

C.1.5 Optimal Step Size

The optimal step size is determined by the following minimization,

$$\alpha_{\min} = \arg\min_{\alpha} \{ M[\mathbf{HR} + \alpha \mathbf{g}, T, \lambda] \}.$$

Expanding $M[\mathbf{HR}+\alpha\mathbf{g},T,\lambda]$ as a Taylor series around $\alpha=0$ results in,

$$\begin{aligned} M[\mathbf{HR}+\alpha\mathbf{g},T,\lambda] &= \sum_{x=0}^{X_{HR}-1}\sum_{y=0}^{Y_{HR}-1}\sum_{c=0}^{3}\rho_T(\mathbf{d}^t_{x,y,c}(\mathbf{HR}+\alpha\mathbf{g})) \\ &\quad +\lambda\sum_{x=0}^{X_{LR}-1}\sum_{y=0}^{Y_{LR}-1}\Bigg(y[x,y] \\ &\quad -\frac{1}{M_xM_y}\sum_{m=xM_x}^{(x+1)M_x-1}\sum_{n=yM_y}^{(y+1)M_y-1}(HR[m,n]+\alpha g[m,n])\Bigg)^2 \\ &= M[\mathbf{HR},T,\lambda]+\frac{\partial}{\partial\alpha}M[\mathbf{HR}+\alpha\mathbf{g},T,\lambda]\Big|_{\alpha=0}\alpha \\ &\quad +\frac{1}{2}\frac{\partial^2}{\partial\alpha^2}M[\mathbf{HR}+\alpha\mathbf{g},T,\lambda]\Big|_{\alpha=0}\alpha^2+\cdots \end{aligned}$$

Taking the derivative of the Taylor series with respect to α and setting it equal to zero results in,

$$\alpha_{\min}=-\left.\frac{\frac{\partial}{\partial\alpha}M[\mathbf{HR}+\alpha\mathbf{g},T,\lambda]}{\frac{\partial^2}{\partial\alpha^2}M[\mathbf{HR}+\alpha\mathbf{g},T,\lambda]}\right|_{\alpha=0}.$$

The numerator of the above is given by,

$$\begin{aligned} \frac{\partial}{\partial\alpha}M[\mathbf{HR}+\alpha\mathbf{g},T,\lambda] &= \sum_{x=0}^{X_{HR}-1}\sum_{y=0}^{Y_{HR}-1}\sum_{c=0}^{3}\rho'_T(\mathbf{d}^t_{x,y,c}(\mathbf{HR}+\alpha\mathbf{g}))(\mathbf{d}^t_{x,y,c}\mathbf{g}) \\ &\quad +2\lambda\sum_{x=0}^{X_{LR}-1}\sum_{y=0}^{Y_{LR}-1}\Bigg[\Bigg(y[x,y] \\ &\quad -\frac{1}{M_xM_y}\sum_{m=xM_x}^{(x+1)M_x-1}\sum_{n=yM_y}^{(y+1)M_y-1}(HR[m,n]+\alpha g[m,n])\Bigg) \\ &\quad \cdot\Bigg(-\frac{1}{M_xM_y}\sum_{m=x}^{(x+1)M_x-1}\sum_{n=yM_y}^{(y+1)M_y-1}g[m,n]\Bigg)\Bigg] \end{aligned}$$

and the denominator is,

$$\frac{\partial^2}{\partial\alpha^2}M[\mathbf{HR}+\alpha\mathbf{g},T,\lambda] = \sum_{x=0}^{X_{HR}-1}\sum_{y=0}^{Y_{HR}-1}\sum_{c=0}^{3}\rho_T''(\mathbf{d}_{x,y,c}^t(\mathbf{HR}+\alpha\mathbf{g}))(\mathbf{d}_{x,y,c}^t\mathbf{g})^2 + 2\lambda\sum_{x=0}^{X_{LR}-1}\sum_{y=0}^{Y_{LR}-1}\left(\frac{1}{M_xM_y}\sum_{m=xM_x}^{(x+1)M_x-1}\sum_{n=yM_y}^{(y+1)M_y-1}g[m,n]\right)^2,$$

where

$$\rho_T''(x)=\begin{cases}2, & |x|\le T\\ 0, & |x|>T\end{cases}.$$

C.1.6 Correction

The HR estimate at the next iteration is given by,

$$\mathbf{HR}^{(it+1)}=\mathbf{HR}^{(it)}+\alpha\mathbf{g},$$

provided that $\alpha^2\|\mathbf{g}\|^2 > e$, where e is some specified error threshold, it is the current iteration and $it+1$ is the next iteration.

C.2 Implementation

The Bayesian algorithm can be implemented by the following steps:

1. Determine the dimensions of the HR estimate using

$$\begin{aligned}X_{HR} &= (M_x-1)(X_{LR}-1)+X_{LR},\\ Y_{HR} &= (M_y-1)(Y_{LR}-1)+Y_{LR}.\end{aligned}$$

2. If an initial condition for the HR estimate has not been specified, perform zero order hold interpolation on the LR image. This is accomplished by up-sampling the LR image by M_x in the horizontal direction and M_y in the vertical direction, creating

an image with Y_{HR} rows and X_{HR} columns of all zeros except at the locations of the original LR samples. The up-sampled image is then filtered with a convolution kernel of all ones, h_0, with M_y rows and M_x columns.

$$\begin{aligned} \uparrow LR[x,y] &= \begin{cases} LR[\frac{x}{M_x}, \frac{y}{M_y}],\ x = 0, M_x, ..X_{HR} - 1, y = 0, M_y, ..Y_{HR} - 1 \\ 0,\ otherwise \end{cases}, \\ HR[x,y] &= h_0[x,y] * \uparrow LR[x,y]. \end{aligned}$$

3. For each iteration, it, of the algorithm:

   (a) Filter the current HR estimate with each of the four clique smoothness measures.

$$\begin{aligned} d_0HR[x,y] &= d_0[x,y] * HR[x,y], \\ d_1HR[x,y] &= d_1[x,y] * HR[x,y], \\ d_2HR[x,y] &= d_2[x,y] * HR[x,y], \\ d_3HR[x,y] &= d_3[x,y] * HR[x,y]. \end{aligned}$$

   (b) Filter the current HR estimate with a moving average filter, h_1, of M_y rows and M_x columns to simulate the sensor model. Down-sample the filtered image by M_x in the horizontal direction and M_y in the vertical direction. Take the pointwise substraction of the LR image by the decimated HR estimate (performed on the LR grid).

$$\begin{aligned} DHR[x,y] &= h_1[x,y] * HR[x,y], \\ \downarrow DHR[x,y] &= DHR[xM_x, yM_y], x = 0, 1, ..X_{LR} - 1, y = 0, 1, ..Y_{LR} - 1, \\ LR\_DHR[x,y] &= LR[x,y] - \downarrow DHR[x,y]. \end{aligned}$$

   (c) The value of the objective function can be computed by applying the Huber

function to each four clique smoothness measures, summing across all terms, and then adding to that the sum of squares of the difference of the LR image and the decimated HR estimate, scaled by the regularization parameter λ.

$$M^{(it)}[\mathbf{HR}, T, \lambda] = \sum_{x=0}^{X_{HR}-1} \sum_{y=0}^{Y_{HR}-1} \sum_{c=0}^{3} \rho_T(d_c HR[x,y]) + \lambda \sum_{x=0}^{X_{LR}-1} \sum_{y=0}^{Y_{LR}-1} LR\_DHR[x,y]^2$$

(d) Compute the smoothing term's contribution to the gradient by applying the first-order derivative of the Huber function to each of the four clique smoothness measures separately, then filtering each one by their respective clique smoothness measures.

$$\begin{aligned} g_0[x,y] &= d_0[x,y] * \rho_T'(d_0 HR[x,y]), \\ g_1[x,y] &= d_1[x,y] * \rho_T'(d_1 HR[x,y]), \\ g_2[x,y] &= d_2[x,y] * \rho_T'(d_2 HR[x,y]), \\ g_3[x,y] &= d_3[x,y] * \rho_T'(d_3 HR[x,y]). \end{aligned}$$

(e) Compute the data fidelity term's contribution to the gradient by applying the sensor model filter to interpolate the difference of the LR image and the decimated HR estimate.

$$\begin{aligned} \uparrow LR\_DHR[x,y] &= \begin{cases} LR\_DHR[\frac{x}{M_x}, \frac{y}{M_y}], \; x = 0, M_x, ..X_{HR}-1, y = 0, M_y, ..Y_{HR}-1 \\ 0, \; otherwise \end{cases} \\ LR\_DHR_1[x,y] &= h_1[x,y] * \uparrow LR\_DHR[x,y]. \end{aligned}$$

(f) Compute the gradient by performing a point-wise sum of the contributions made by the smoothing term and the data fidelity term (performed on the HR

grid).

$$g[x,y] = \sum_{c=0}^{3} g_c[x,y] - 2\lambda LR\_DHR_1[x,y].$$

(g) Compute the smoothing term's contribution to the numerator and denominator of the optimal step size.

$$\begin{aligned} d_0g[x,y] &= d_0[x,y] * g[x,y], \\ d_1g[x,y] &= d_1[x,y] * g[x,y], \\ d_2g[x,y] &= d_2[x,y] * g[x,y], \\ d_3g[x,y] &= d_3[x,y] * g[x,y], \end{aligned}$$

$$\begin{aligned} \alpha_n s &= \sum_{x=0}^{X_{HR}-1} \sum_{y=0}^{Y_{HR}-1} \sum_{c=0}^{3} \rho_T{}'(d_cHR[x,y]) d_cg[x,y], \\ \alpha_d s &= \sum_{x=0}^{X_{HR}-1} \sum_{y=0}^{Y_{HR}-1} \sum_{c=0}^{3} \rho_T{}''(d_cHR[x,y]) d_cg[x,y]^2. \end{aligned}$$

(h) Compute the data fidelity term's contribution to the optimal step size.

$$\begin{aligned} Dg[x,y] &= h_1[x,y] * g[x,y], \\ \downarrow Dg[x,y] &= Dg[xM_x, yM_y], x = 0,1,..,X_{LR}-1, y = 0,1,..,Y_{LR}-1, \end{aligned}$$

$$\begin{aligned} \alpha_n d &= \sum_{x=0}^{X_{LR}-1} \sum_{y=0}^{Y_{LR}-1} LR\_DHR[x,y] \downarrow Dg[x,y], \\ \alpha_d d &= \sum_{x=0}^{X_{LR}-1} \sum_{y=0}^{Y_{LR}-1} \downarrow Dg[x,y]^2. \end{aligned}$$

(i) Compute the optimal step size.

$$\alpha = -\frac{\alpha_n s - 2\lambda \alpha_n d}{\alpha_d s + 2\lambda \alpha_d d}.$$

(j) Test the termination condition $\|\alpha \mathbf{g}\|^2$. If the condition is less than the specified error threshold, then terminate the iterations. Otherwise, move the HR estimate closer to the optimal solution.

$$\mathbf{HR}^{(it+1)} = \mathbf{HR}^{(it)} + \alpha \mathbf{g}.$$

Appendix D

Detailed Description of the Improved Statistical Algorithm

This appendix provides a detailed description of the theory and implementation of the improved statistical magnification algorithm proposed in Section 4.2.

D.1 Theory

D.1.1 Minimization Problem

The improved Bayesian estimation problem reduces to the following minimization,

$$\mathbf{HR}_{\min} = \arg\min_{\mathbf{HR}}\{M[\mathbf{HR}, \lambda_1, \lambda_2]\}.$$

The objective function is,

$$
\begin{aligned}
M[\mathbf{HR}, \lambda_1, \lambda_2] &= \Omega[\mathbf{HR}, \lambda_1, \lambda_2] + |\mathbf{LR} - \mathbf{D}(\mathbf{HR})\|^2 \\
&= \sum_{x=0}^{X_{HR}-1} \sum_{y=0}^{Y_{HR}-1} \sum_{c=0}^{3} w_c[x, y, \lambda_1, \lambda_2](\mathbf{d}_{x,y,c}^t \mathbf{HR})^2 \\
&\quad + \sum_{x=0}^{X_{LR}-1} \sum_{y=0}^{Y_{LR}-1} \left(y[x, y] - \frac{1}{M_x M_y} \sum_{m=xM_x}^{(x+1)M_x-1} \sum_{n=yM_y}^{(y+1)M_y-1} HR[m, n] \right)^2
\end{aligned}
$$

where **LR** is a lexicographically ordered $X_{LR}Y_{LR} \times 1$ vector of samples from the original low-resolution image, **HR** is the $X_{HR}Y_{HR} \times 1$ vector of samples from the high-resolution estimate, **D** is a $X_{LR}Y_{LR} \times X_{HR}Y_{HR}$ constant block-diagonal decimation matrix and $w_c[x, y, \lambda_1, \lambda_2]$ are the weights applied to each directional smoothing component.

Note that the dimensions of the HR estimate are such that $X_{HR} = (M_x - 1)(X_{LR} - 1) + X_{LR}$ and $Y_{HR} = (M_y - 1)(Y_{LR} - 1) + Y_{LR}$, where M_x and M_y are the magnification factors in the horizontal and vertical directions, and X_{LR} and Y_{LR} are the horizontal and vertical dimensions of the LR image.

The cliques are defined on second order neighborhoods with the following potentials:

$$
\begin{aligned}
1)\ \mathbf{d}_{x,y,0}^t \mathbf{HR} &= HR[x-1, y] - 2HR[x, y] + HR[x+1, y], \\
2)\ \mathbf{d}_{x,y,1}^t \mathbf{HR} &= HR[x-1, y+1] - 2HR[x, y] + HR[x+1, y-1], \\
3)\ \mathbf{d}_{x,y,2}^t \mathbf{HR} &= HR[x, y-1] - 2HR[x, y] + HR[x, y+1], \\
4)\ \mathbf{d}_{x,y,3}^t \mathbf{HR} &= HR[x-1, y-1] - 2HR[x, y] + HR[x+1, y+1],
\end{aligned}
$$

where $HR[x, y]$ is the value of the sample at coordinates (x, y). Note that the evaluation of the clique smoothness measures are equivalent to preforming 2D filtering operations.

The convolution kernel for each clique is therefore,

$$d_0 = \begin{bmatrix} 0 & 0 & 0 \\ 1 & -2 & 1 \\ 0 & 0 & 0 \end{bmatrix}, d_1 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & -2 & 0 \\ 1 & 0 & 0 \end{bmatrix}, d_2 = \begin{bmatrix} 0 & 1 & 0 \\ 0 & -2 & 0 \\ 0 & 1 & 0 \end{bmatrix}, d_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -2 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

D.1.2 Gradient of the Objective Function

The first step in the optimization procedure is to compute the gradient of the objective function $M[\mathbf{HR}, \lambda_1, \lambda_2]$ with respect to every sample in the high-resolution estimate. Let this gradient be defined as,

$$\begin{aligned} \mathbf{g} &= \nabla M[\mathbf{HR}, \lambda_1, \lambda_2] \\ &= \nabla \Omega[\mathbf{HR}, \lambda_1, \lambda_2] + \nabla \|\mathbf{LR} - \mathbf{D}(\mathbf{HR})\|^2 \\ &= \mathbf{g}_1 + \mathbf{g}_2 \end{aligned}$$

where,

$$\nabla = \sum_{x=0}^{X_{HR}-1} \sum_{y=0}^{Y_{HR}-1} \hat{\mathbf{HR}}_{x,y} \frac{\partial}{\partial HR_{x,y}},$$

$\hat{\mathbf{HR}}_{x,y}$ is the unit vector which is one at sample $HR[x, y]$ and zero everywhere else, $\frac{\partial}{\partial HR_{x,y}}$ is the partial derivative taken with respect to sample $HR[x, y]$, $\mathbf{g}_1$ is the gradient of the smoothing term and $\mathbf{g}_2$ is the gradient of the data fidelity term.

D.1.3 Gradient of the Smoothing Term

The gradient of the smoothing term is given by

$$\begin{aligned} \mathbf{g}_1 &= \nabla \Omega[\mathbf{HR}, \lambda_1, \lambda_2] \\ &= \sum_{x=0}^{X_{HR}-1} \sum_{y=0}^{Y_{HR}-1} \hat{\mathbf{HR}}_{x,y} \frac{\partial}{\partial HR_{x,y}} \Omega[\mathbf{HR}, \lambda_1, \lambda_2]. \end{aligned}$$

After expanding the summations in $\Omega[\mathbf{HR}, \lambda_1, \lambda_2]$, the terms containing $HR[x, y]$ are:

$$
\begin{aligned}
1)\quad & \mathbf{d}^t_{x,y,0}\mathbf{HR} = HR[x-1,y] - 2HR[x,y] + HR[x+1,y], \\
2)\quad & \mathbf{d}^t_{x,y,1}\mathbf{HR} = HR[x-1,y+1] - 2HR[x,y] + HR[x+1,y-1], \\
3)\quad & \mathbf{d}^t_{x,y,2}\mathbf{HR} = HR[x,y-1] - 2HR[x,y] + HR[x,y+1], \\
4)\quad & \mathbf{d}^t_{x,y,3}\mathbf{HR} = HR[x-1,y-1] - 2HR[x,y] + HR[x+1,y+1], \\
5)\quad & \mathbf{d}^t_{x-1,y-1,3}\mathbf{HR} = HR[x-2,y-2] - 2HR[x-1,y-1] + HR[x,y], \\
6)\quad & \mathbf{d}^t_{x,y-1,2}\mathbf{HR} = HR[x,y-2] - 2HR[x,y-1] + HR[x,y], \\
7)\quad & \mathbf{d}^t_{x+1,y-1,1}\mathbf{HR} = HR[x,y] - 2HR[x+1,y-1] + HR[x+2,y-2], \\
8)\quad & \mathbf{d}^t_{x-1,y,0}\mathbf{HR} = HR[x-2,y] - 2HR[x-1,y] + HR[x,y], \\
9)\quad & \mathbf{d}^t_{x+1,y,0}\mathbf{HR} = HR[x,y] - 2HR[x+1,y] + HR[x+2,y], \\
10)\quad & \mathbf{d}^t_{x-1,y+1,1}\mathbf{HR} = HR[x-2,y+2] - 2HR[x-1,y+1] + HR[x,y], \\
11)\quad & \mathbf{d}^t_{x,y+1,2}\mathbf{HR} = HR[x,y] - 2HR[x,y+1] + HR[x,y+2], \\
12)\quad & \mathbf{d}^t_{x+1,y+1,3}\mathbf{HR} = HR[x,y] - 2HR[x+1,y+1] + HR[x+2,y+2].
\end{aligned}
$$

Therefore, the first-order partial derivative of $\Omega[\mathbf{HR}, \lambda_1, \lambda_2]$ with respect to each high-

resolution sample is,

$$\begin{aligned}
\frac{\partial}{\partial HR_{x,y}}\Omega[\mathbf{HR},\lambda_1,\lambda_2] &= \frac{\partial}{\partial HR_{x,y}}\sum_{m=-1}^{1}\Big[w_0[x+m,y,\lambda_1,\lambda_2](\mathbf{d}^t_{(x+m),y,0}\mathbf{HR})^2 \\
&\quad +w_1[x-m,y+m,\lambda_1,\lambda_2](\mathbf{d}^t_{(x-m),(y+m),1}\mathbf{HR})^2 \\
&\quad +w_2[x,y+m,\lambda_1,\lambda_2](\mathbf{d}^t_{x,(y+m),2}\mathbf{HR})^2 \\
&\quad +w_3[x+m,y+m,\lambda_1,\lambda_2](\mathbf{d}^t_{(x+m),(y+m),3}\mathbf{HR})^2\Big] \\
&= 2\sum_{m=-1}^{1}\Big[d_0[x+m,y]w_0[x+m,y,\lambda_1,\lambda_2](\mathbf{d}^t_{(x+m),y,0}\mathbf{HR}) \\
&\quad +d_1[x-m,y+m]w_1[x-m,y+m,\lambda_1,\lambda_2] \\
&\quad \cdot(\mathbf{d}^t_{(x-m),(y+m),1}\mathbf{HR}) \\
&\quad +d_2[x,y+m]w_2[x,y+m,\lambda_1,\lambda_2](\mathbf{d}^t_{x,(y+m),2}\mathbf{HR}) \\
&\quad +d_3[x+m,y+m]w_3[x+m,y+m,\lambda_1,\lambda_2] \\
&\quad \cdot(\mathbf{d}^t_{(x+m),(y+m),3}\mathbf{HR})\Big]
\end{aligned}$$

From the above, the contribution of each clique to the gradient is simply the HR estimate filtered by each of the clique smoothness measures, followed by the application of the directional weights, followed by another filtering by the clique smoothness measures.

D.1.4 Gradient of the Data Fidelity Term

The gradient of the data fidelity term is given by

$$\begin{aligned}
\mathbf{g}_2 &= \nabla\|\mathbf{LR}-\mathbf{D}(\mathbf{HR})\|^2 \\
&= \sum_{x=0}^{X_{HR}-1}\sum_{y=0}^{Y_{HR}-1}\hat{\mathbf{HR}}_{x,y}\frac{\partial}{\partial HR_{x,y}}\|\mathbf{LR}-\mathbf{D}(\mathbf{HR})\|^2.
\end{aligned}$$

After expanding $\|\mathbf{LR}-\mathbf{D}(\mathbf{HR})\|^2$, there is only one term containing $HR[x,y]$. Therefore, the first-order partial derivative of $\|\mathbf{LR}-\mathbf{D}(\mathbf{HR})\|^2$ with respect to each high-resolution

sample is,

$$\frac{\partial}{\partial HR_{x,y}}\|\mathbf{LR}-\mathbf{D}(\mathbf{HR})\|^2 = 2\left(LR[i,j] - \frac{1}{M_xM_y}\sum_{m=iM_x}^{(i+1)M_x-1}\sum_{n=jM_y}^{(j+1)M_y-1} HR[m,n]\right)\left(-\frac{1}{M_xM_y}\right),$$

where $i = \left\lfloor \frac{x}{M_x} \right\rfloor$ and $j = \left\lfloor \frac{y}{M_y} \right\rfloor$.

D.1.5 Optimal Step Size

The optimal step size is determined by the following minimization,

$$\alpha_{\min} = \arg\min_{\alpha}\{M[\mathbf{HR}+\alpha\mathbf{g},\lambda_1,\lambda_2]\}.$$

Expanding $M[\mathbf{HR}+\alpha\mathbf{g},\lambda_1,\lambda_2]$ as a Taylor series around $\alpha = 0$ results in,

$$\begin{aligned} M[\mathbf{HR}+\alpha\mathbf{g},\lambda_1,\lambda_2] &= \sum_{x=0}^{X_{HR}-1}\sum_{y=0}^{Y_{HR}-1}\sum_{c=0}^{3} w_c[x,y,\lambda_1,\lambda_2](\mathbf{d}^t_{x,y,c}(\mathbf{HR}+\alpha\mathbf{g}))^2 \\ &\quad + \sum_{x=0}^{X_{LR}-1}\sum_{y=0}^{Y_{LR}-1}\left(y[x,y]\right. \\ &\quad \left. -\frac{1}{M_xM_y}\sum_{m=xM_x}^{(x+1)M_x-1}\sum_{n=yM_y}^{(y+1)M_y-1}(HR[m,n]+\alpha g[m,n])\right)^2 \\ &= M[\mathbf{HR},\lambda_1,\lambda_2] + \frac{\partial}{\partial\alpha}M[\mathbf{HR}+\alpha\mathbf{g},\lambda_1,\lambda_2]\Big|_{\alpha=0}\alpha \\ &\quad + \frac{1}{2}\frac{\partial^2}{\partial\alpha^2}M[\mathbf{HR}+\alpha\mathbf{g},\lambda_1,\lambda_2]\Big|_{\alpha=0}\alpha^2+\cdots \end{aligned}$$

Taking the derivative of the Taylor series with respect to α and setting it equal to zero results in,

$$\alpha_{\min} = -\left.\frac{\frac{\partial}{\partial\alpha}M[\mathbf{HR}+\alpha\mathbf{g},\lambda_1,\lambda_2]}{\frac{\partial^2}{\partial\alpha^2}M[\mathbf{HR}+\alpha\mathbf{g},\lambda_1,\lambda_2]}\right|_{\alpha=0}.$$

The numerator of the above is given by,

$$\begin{aligned}\frac{\partial}{\partial\alpha}M[\mathbf{HR}+\alpha\mathbf{g},\lambda_1,\lambda_2] &= 2\sum_{x=0}^{X_{HR}-1}\sum_{y=0}^{Y_{HR}-1}\sum_{c=0}^{3}w_c[x,y,\lambda_1,\lambda_2](\mathbf{d}_{x,y,c}^t(\mathbf{HR}+\alpha\mathbf{g}))(\mathbf{d}_{x,y,c}^t\mathbf{g}) \\ &\quad +2\sum_{x=0}^{X_{LR}-1}\sum_{y=0}^{Y_{LR}-1}\left[\left(y[x,y]\right.\right. \\ &\quad \left.-\frac{1}{M_xM_y}\sum_{m=xM_x}^{(x+1)M_x-1}\sum_{n=yM_y}^{(y+1)M_y-1}(HR[m,n]+\alpha g[m,n])\right) \\ &\quad \left.\cdot\left(-\frac{1}{M_xM_y}\sum_{m=x}^{(x+1)M_x-1}\sum_{n=yM_y}^{(y+1)M_y-1}g[m,n]\right)\right]\end{aligned}$$

and the denominator is,

$$\begin{aligned}\frac{\partial^2}{\partial\alpha^2}M[\mathbf{HR}+\alpha\mathbf{g},\lambda_1,\lambda_2] &= 2\sum_{x=0}^{X_{HR}-1}\sum_{y=0}^{Y_{HR}-1}\sum_{c=0}^{3}w_c[x,y,\lambda_1,\lambda_2](\mathbf{d}_{x,y,c}^t\mathbf{g})^2 \\ &\quad +2\sum_{x=0}^{X_{LR}-1}\sum_{y=0}^{Y_{LR}-1}\left(\frac{1}{M_xM_y}\sum_{m=xM_x}^{(x+1)M_x-1}\sum_{n=yM_y}^{(y+1)M_y-1}g[m,n]\right)^2.\end{aligned}$$

D.1.6 Correction

The correction used by the improved algorithm is unchanged. Refer to Section C.1.6 of Appendix C for further details.

D.2 Implementation

The improved Bayesian algorithm can be implemented by the following steps:

1. Determine the dimensions of the HR estimate using

$$\begin{aligned}X_{HR} &= (M_x-1)(X_{LR}-1)+X_{LR}, \\ Y_{HR} &= (M_y-1)(Y_{LR}-1)+Y_{LR}.\end{aligned}$$

2. If an initial condition for the HR estimate has not been specified, perform zero order hold interpolation on the LR image. This is accomplished by up-sampling the LR image by M_x in the horizontal direction and M_y in the vertical direction, creating an image with Y_{HR} rows and X_{HR} columns of all zeros except at the locations of the original LR samples. The up-sampled image is then filtered with a convolution kernel of all ones, h_0, with M_y rows and M_x columns.

$$\begin{aligned} \uparrow LR[x,y] &= \begin{cases} LR[\frac{x}{M_x},\frac{y}{M_y}],\ x=0,M_x,..X_{HR}-1, y=0,M_y,..Y_{HR}-1 \\ 0,\ otherwise \end{cases}, \\ HR[x,y] &= h_0[x,y]* \uparrow LR[x,y]. \end{aligned}$$

3. If an initial condition for the estimated binary HR edge map, $e[x,y]$, has not been specified, perform the sub-pixel edge estimation process described in Section B.1.

   From the binary edge map, determine the weights, $w_c[x,y,\lambda_1,\lambda_2]$, for each of the four cliques by applying the following procedure:

$$\begin{aligned} e_0[x,y] &= |d_0[x,y]*e[x,y]|>0, \\ e_1[x,y] &= |d_1[x,y]*s(e[x,y])|>0, \\ e_2[x,y] &= |d_2[x,y]*e[x,y]|>0, \\ e_3[x,y] &= |d_3[x,y]*s(e[x,y])|>0, \end{aligned}$$

$$w_c[x,y,\lambda_1,\lambda_2] = (\lambda_2-\lambda_1)e_c[x,y]+\lambda_1,$$

   where $s(\cdot)$ is the morphological operator used for spreading edges. The morphological operator is implemented by a simple binary pattern matching algorithm. The binary edge map is searched for groups of four samples that occur in the following two patterns: $\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ and $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$. Matches are then replaced by $\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$.

4. For each iteration, it, of the algorithm:

   (a) Filter the current HR estimate with each of the four clique smoothness measures.

$$\begin{aligned} d_0HR[x,y] &= d_0[x,y] * HR[x,y], \\ d_1HR[x,y] &= d_1[x,y] * HR[x,y], \\ d_2HR[x,y] &= d_2[x,y] * HR[x,y], \\ d_3HR[x,y] &= d_3[x,y] * HR[x,y]. \end{aligned}$$

   (b) Filter the current HR estimate with a moving average filter, h_1, of M_y rows and M_x columns to simulate the sensor model. Down-sample the filtered image by M_x in the horizontal direction and M_y in the vertical direction. Take the pointwise substraction of the LR image by the decimated HR estimate (performed on the LR grid).

$$\begin{aligned} DHR[x,y] &= h_1[x,y] * HR[x,y], \\ \downarrow DHR[x,y] &= DHR[xM_x, yM_y], x = 0,1,..X_{LR}-1, y = 0,1,..Y_{LR}-1, \\ LR\_DHR[x,y] &= LR[x,y] - \downarrow DHR[x,y]. \end{aligned}$$

   (c) The value of the objective function can be computed by squaring each four clique smoothness measures, summing across all terms, and then adding to that the sum of squares of the difference of the LR image and the decimated HR estimate, scaled by the regularization parameter λ.

$$\begin{aligned} M^{(it)}[\mathbf{HR}, \lambda_1, \lambda_2] = & \sum_{x=0}^{X_{HR}-1} \sum_{y=0}^{Y_{HR}-1} \sum_{c=0}^{3} w_c[x,y,\lambda_1,\lambda_2] d_c HR[x,y]^2 \\ & + \sum_{x=0}^{X_{LR}-1} \sum_{y=0}^{Y_{LR}-1} LR\_DHR[x,y]^2 \end{aligned}$$

(d) Compute the smoothing term's contribution to the gradient by applying the first-order derivative of the Huber function to each of the four clique smoothness measures separately, then filtering each one by their respective clique smoothness measures.

$$\begin{aligned} g_0[x,y,\lambda_1,\lambda_2] &= d_0[x,y] * (w_0[x,y,\lambda_1,\lambda_2] d_0 HR[x,y]), \\ g_1[x,y,\lambda_1,\lambda_2] &= d_1[x,y] * (w_1[x,y,\lambda_1,\lambda_2] d_1 HR[x,y]), \\ g_2[x,y,\lambda_1,\lambda_2] &= d_2[x,y] * (w_2[x,y,\lambda_1,\lambda_2] d_2 HR[x,y]), \\ g_3[x,y,\lambda_1,\lambda_2] &= d_3[x,y] * (w_3[x,y,\lambda_1,\lambda_2] d_3 HR[x,y]). \end{aligned}$$

(e) Compute the data fidelity term's contribution to the gradient by applying the sensor model filter to interpolate the difference of the LR image and the decimated HR estimate.

$$\begin{aligned} \uparrow LR\_DHR[x,y] &= \begin{cases} LR\_DHR[\frac{x}{M_x}, \frac{y}{M_y}],\ x = 0, M_x, ..X_{HR} - 1, y = 0, M_y, ..Y_{HR} - 1 \\ 0,\ otherwise \end{cases} \\ LR\_DHR_1[x,y] &= h_1[x,y] * \uparrow LR\_DHR[x,y]. \end{aligned}$$

(f) Compute the gradient by performing a point-wise sum of the contributions made by the smoothing term and the data fidelity term (performed on the HR grid).

$$g[x,y,\lambda_1,\lambda_2] = 2\left[\sum_{c=0}^{3} g_c[x,y,\lambda_1,\lambda_2] - LR\_DHR_1[x,y]\right].$$

(g) Compute the smoothing term's contribution to the numerator and denominator

of the optimal step size.

$$
\begin{aligned}
d_0 g[x, y, \lambda_1, \lambda_2] &= d_0[x, y] * g[x, y, \lambda_1, \lambda_2], \\
d_1 g[x, y, \lambda_1, \lambda_2] &= d_1[x, y] * g[x, y, \lambda_1, \lambda_2], \\
d_2 g[x, y, \lambda_1, \lambda_2] &= d_2[x, y] * g[x, y, \lambda_1, \lambda_2], \\
d_3 g[x, y, \lambda_1, \lambda_2] &= d_3[x, y] * g[x, y, \lambda_1, \lambda_2],
\end{aligned}
$$

$$
\begin{aligned}
\alpha_n s[\lambda_1, \lambda_2] &= \sum_{x=0}^{X_{HR}-1} \sum_{y=0}^{Y_{HR}-1} \sum_{c=0}^{3} w_c[x, y, \lambda_1, \lambda_2] d_c HR[x, y] d_c g[x, y, \lambda_1, \lambda_2], \\
\alpha_d s[\lambda_1, \lambda_2] &= \sum_{x=0}^{X_{HR}-1} \sum_{y=0}^{Y_{HR}-1} \sum_{c=0}^{3} w_c[x, y, \lambda_1, \lambda_2] d_c g[x, y, \lambda_1, \lambda_2]^2.
\end{aligned}
$$

(h) Compute the data fidelity term's contribution to the optimal step size.

$$
\begin{aligned}
Dg[x, y] &= h_1[x, y] * g[x, y], \\
\downarrow Dg[x, y] &= Dg[xM_x, yM_y], x = 0, 1, .., X_{LR} - 1, y = 0, 1, .., Y_{LR} - 1,
\end{aligned}
$$

$$
\begin{aligned}
\alpha_n d &= \sum_{x=0}^{X_{LR}-1} \sum_{y=0}^{Y_{LR}-1} LR\_DHR[x, y] \downarrow Dg[x, y], \\
\alpha_d d &= \sum_{x=0}^{X_{LR}-1} \sum_{y=0}^{Y_{LR}-1} \downarrow Dg[x, y]^2.
\end{aligned}
$$

(i) Compute the optimal step size.

$$
\alpha = -\frac{\alpha_n s[\lambda_1, \lambda_2] - \alpha_n d}{\alpha_d s[\lambda_1, \lambda_2] + \alpha_d d}.
$$

(j) Test the termination condition $\|\alpha \mathbf{g}\|^2$. If the condition is less than the specified error threshold, then terminate the iterations. Otherwise, move the HR

estimate closer to the optimal solution.

$$\mathbf{HR}^{(it+1)} = \mathbf{HR}^{(it)} + \alpha \mathbf{g}.$$

References

[1] T. M. Lehmann, C. Gönner, and K. Spitzer, "Survey: Interpolation methods in medical image processing," *IEEE Trans. Medical Imaging*, vol. 18, pp. 1049–1075, November 1999.

[2] A. Gotchev, J. Vesma, T. Saramäki, and K. Egiazarian, "Digital image resampling by modified B-spline functions," *IEEE Nordic Signal Processing Symposium*, pp. 259–262, June 2000.

[3] H. Chen and G. E. Ford, "An FIR interpolation filter design method based on properties of human vision," *Proc. IEEE Int. Conf. Image Processing*, vol. 3, pp. 581–585, November 1994.

[4] S. E. Reichenbach and F. Geng, "Two-dimensional cubic convolution," *IEEE Trans. Image Process.*, vol. 12, pp. 857–865, August 2003.

[5] H. Jiang and C. Moloney, "A new direction adaptive scheme for image interpolation," *Proc. IEEE Int. Conf. Image Processing*, vol. 3, pp. 369–372, 2002.

[6] B. S. Morse and D. Schwartzwald, "Image magnification using level-set reconstruction," *Proc. IEEE Conf. Computer Vision Pattern Recognition*, vol. 3, pp. 333–340, 2001.

[7] H. Aly and E. Dubois, "Regularized image up-sampling using a new observation model and the level set method," *Proc. IEEE Int. Conf. Image Processing*, vol. 3, pp. 665–668, September 2003.

[8] Y. Takahashi and A. Taguchi, "An enlargement method of digital images with the prediction of high-frequency components," *Proc. IEEE Int. Conf. Acoustics Speech Signal Processing*, vol. 4, pp. 3700–3703, 2002.

[9] X. Lu, P. S. Hong, and M. J. T. Smith, "An efficient directional image interpolation method," *Proc. IEEE Int. Conf. Acoustics Speech Signal Processing*, vol. 3, pp. 97–100, 2003.

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

[11] K. Kinebuchi, D. D. Muresan, and T. W. Parks, "Image interpolation using wavelet-based hidden Markov trees," *Proc. IEEE Int. Conf. Acoustics Speech Signal Processing*, vol. 3, pp. 1957–1960, 2001.

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

[13] D. Rajan and S. Chaudhuri, "A perceptually organized method for image interpolation," *Proc. IEEE Int. Conf. Pattern Recognition*, vol. 3, pp. 734–737, 2000.

[14] W. K. Carey, D. B. Chuang, and S. S. Hemami, "Regularity-preserving image interpolation," *IEEE Trans. Image Process.*, vol. 8, pp. 1293–1297, September 1999.

[15] L. S. DeBrunner and V. DeBrunner, "Color image interpolation with edge-enhancement," *Proc. IEEE Conf. Signals, Systems and Computers*, vol. 2, pp. 901–905, 2000.

[16] A. Biancardi, L. Cinque, and L. Lombardi, "Improvements to image magnification," *Pattern Recognit.*, vol. 35, pp. 677–687, March 2002.

[17] A. M. Darwish, M. S. Bedair, and S. I. Shaheen, "Adaptive resampling algorithm for image zooming," *IEE Proc., Vis., Image and Signal Process.*, vol. 144, pp. 207–212, August 1997.

[18] X. Li and M. T. Orchard, "New edge-directed interpolation," *IEEE Trans. Image Process.*, vol. 10, pp. 1521–1527, October 2001.

[19] J. Allebach and P. W. Wong, "Edge-directed interpolation," *Proc. IEEE Int. Conf. Image Processing*, vol. 3, pp. 707–710, September 1996.

[20] H. Shi and R. Ward, "Canny edge based image expansion," *Proc. IEEE Int. Symp. Circuits and Systems*, vol. 1, pp. 785–788, 2002.

[21] F. Arándiga, R. Donat, and P. Mulet, "Adaptive interpolation of images," *Signal Process.*, vol. 83, pp. 459–464, February 2003.

[22] G. Ramponi, "Warped distance for space-variant linear image interpolation," *IEEE Trans. Image Process.*, vol. 8, pp. 629–639, May 1999.

[23] J.-K. Han and S.-U. Baek, "Parametric cubic convolution scaler for enlargement and reduction of image," *IEEE Trans. Consum. Electron.*, vol. 46, pp. 247–256, May 2000.

[24] S. Battiato, G. Gallo, and F. Stanco, "A locally adaptive zooming algorithm for digital images," *Image Vis. Comput.*, vol. 20, pp. 805–812, September 2002.

[25] C.-S. Chuah and J.-J. Leou, "An adaptive image interpolation algorithm for image/video processing," *Pattern Recognit.*, vol. 34, pp. 2383–2393, December 2001.

[26] Q. Wang and R. Ward, "A new edge-directed image expansion scheme," *Proc. IEEE Int. Conf. Image Processing*, vol. 3, pp. 899–902, 2001.

[27] S. Dube and L. Hong, "An adaptive algorithm for image resolution enhancement," *Proc. IEEE Conf. Signals, Systems and Computers*, vol. 2, pp. 1731–1734, 2000.

[28] M. Ohira, K. Mori, K. Wada, and K. Traichi, "High quality image restoration by adaptively transformed sampling function," *Proc. IEEE Conf. Communications, Computers, and Signal Processing*, pp. 201–204, 1999.

[29] R. R. Schultz and R. L. Stevenson, "A Bayesian approach to image expansion for improved definition," *IEEE Trans. Image Process.*, vol. 3, pp. 233–242, May 1994.

[30] C. B. Atkins, C. A. Bouman, and J. P. Allebach, "Optimal image scaling using pixel classification," *Proc. IEEE Int. Conf. Image Processing*, vol. 3, pp. 864–867, 2001.

[31] M. Sakalli, H. Yan, and A. M. N. Fu, "A fuzzy-Bayesian approach to image expansion," *Proc. Int. Joint Conf. Neural Networks*, vol. 4, pp. 2685–2689, 1999.

[32] R. R. Schultz and R. L. Stevenson, "Extraction of high-resolution frames from video sequences," *IEEE Trans. Image Process.*, vol. 5, pp. 996–1011, June 1996.

[33] D. Chen and R. R. Schultz, "Extraction of high-resolution video stills from MPEG image sequences," *Proc. IEEE Int. Conf. Image Processing*, vol. 2, pp. 465–469, October 1998.

[34] H. Aly and E. Dubois, "Crafting the observation model for regularized image up-sampling," *Proc. IEEE Int. Conf. Acoustics Speech Signal Processing*, vol. 3, pp. 101–104, April 2003.

[35] A. Bovik, ed., *Handbokk of Image & Video Processing.* Academic Press, 2000.

[36] *Matlab Image Processing Toolbox Online Help.*

[37] P. W. Wong and J. P. Allebach, "Convergence of an iterative edge-directed image interpolation algorithm," *Proc. IEEE Int. Symp. Circuits and Systems*, pp. 1173–1176, June 1997.

[38] H. A. Aly, *Regularized image up-sampling.* PhD thesis, University of Ottawa, 2004.

[39] R. L. Stevenson, B. E. Schmitz, and E. J. Delp, "Discontinuity preserving regularization of inverse visual problems," *IEEE Trans. Syst. Man Cybern.*, vol. 24, March 1994.