アルゴリズム 78−5
（2001. 5. 18）

Image Interpolation with Edge Enhancement

PeiFeng Zeng,
School of Engineering
Nagoya University
Furo-cho, Chikusa-ku
Nagoya, Japan
zengpf@ieee.org

Tomio Hirata
School of Engineering
Nagoya University
Furo-cho, Chikusa-ku
Nagoya, Japan
hirata@nuee.nagoya-u.ac.jp

Abstract

Interpolation is often used in image processing. Edges are blurred when images are processed with linear interpolations. In this paper, an algorithm for image interpolation with edge enhancement is discussed. To make detail edges in images more perceptible, images are expanded with spline interpolation. Intensity of all newly inserted pixels is determined based on the intensity distribution around it. After interpolation, edges are separated far away and it is easy to do the edge detection in interpolated images. Unlike other edge operators that convolute a smoothing function to images directly, the algorithm discussed in this paper smoothes the image by interpolation in which the image is expanded and smoothed, and thus detailed information remains.

A unified linear-time algorithm for distance transform is used to deal with the calculation of Euclidean distance. So-called edge area (EA) is determined which is located around edges. Erosion is taken place in EA to determine the erode source intensity (ESI).

Based upon distance and ESI, intensity of every pixel is adjusted. Edge enhancement is performed repeatedly. By repeating the edge enhancement, edges in images get clearer.

Keywords: B-spline, interpolation, edge detection, Euclidean distance transform, edge enhancement

1 Introduction

Interpolation is often used in image processing. Because there exist fast algorithms for B-spline interpolation, B-splines are used to perform fast image interpolation. Edges are blurred when images are processed with linear interpolations. When images are out of focus, edges are also appeared to be blurry. In the above two cases edges are widened and turn to be ramps.

Edge sharpening is a classical problem. Among such approaches, one is histogram equalization, in which each intensity level to a new value. It is effective to improve image contrast. In neighborhood operation, difference between opposite sides of edges is enlarged, i.e., the bright side of and edge becomes brighter while dark side gets darker after operation. The shortcoming is overshooting caused in outputs.

Edge sharpening can also be performed in frequency domain. For there exists large amount of high frequency component and edge turns out to be sharper when high frequency components are emphasized. For ramp edges, such processing will cause distortions near their turning points.

Above edge sharpening algorithms mainly increase the contrast across edges, but edges become wider when images are interpolated or out of focus.

In this paper, algorithms for edge enhancement are proposed to improve edges that are blurred by B-spline interpolation and narrow their transient areas.

In practice, we found that it is hard to detect fine texture or edges in images with low resolution, there is little useful information in the results because edges are too near to each other. To separate these edges and make them more meaningful, images with low resolution are interpolated with B-spline transform in this paper. In the interpolation, edges are smoothed but their direction and intensity are closely related to that of edges in input images. So we can get information of ordinary images from these edges easily. The first derivative turns to be continuous after images expanded with interpolation. By shifting the expanding result left and right (up and down), calculating the difference between the shifting results, we get the first derivative along x direction (y direction). With this information, edge distribution and intensity are calculated.

Expansion is implemented based on the detected edges. Distance from non-edge pixels to nearest edge pixels is calculated. Pixels with small distance value are regarded to be in the so-called Edge Area (EA) where edge enhancement will be performed.

Then, erosion is performed in EA. Firstly, all out-

most EA pixels are determined as erosion source, and their intensity to be erosion intensity. After that, Erosion is executed along the deepest direction of distance towards edge pixels, i.e., erosion direction. Pixels in the erosion direction are eroded and turn out to be new erosion sources. Erosion intensity is also transferred in the erosion direction.

Based on the distance and erosion intensity, intensity adjustment is executed repeatedly. Intensity in EA is adjusted and edges get more and more enhanced after each intensity adjustment.

2 Preliminaries

In this paper, B-splines are used to do the interpolation and edge detection, here we list some properties about B-splines and their transform.

2.1 B-Splines

B-spline of order 0 is defined as the central unit pulse function

$$\beta^0(x) = \begin{cases} 1, & -\frac{1}{2} \le x \le \frac{1}{2} \\ 0, & \text{Otherwise} \end{cases}.$$

nth-order B-splines are defined as the convolution of $n-1$th order B-splines with $\beta^0(x)$

$$\beta^n(x) = \beta^0 * \beta^{n-1}(x) = \overbrace{\beta^0 * \beta^0 * \cdots * \beta^0(x)}^{n+1}, \ x \in \mathbf{R}. \quad (1)$$

nth-order B-spline is a positive function defined in $[-\frac{n+1}{2}, \frac{n+1}{2}]$ with $n+2$ equally spaced knots, and it has continuous derivatives up to order $n-1$.

The first derivative of nth-order B-spline

$$\frac{\partial \beta^n(x)}{\partial x} = \beta^{n-1}\left(x + \frac{1}{2}\right) - \beta^{n-1}\left(x - \frac{1}{2}\right) \quad (2)$$

is equal to the subtraction of two translated $n-1$th-order B-splines.

$\beta_m^n(x)$ denotes B-splines expanded by a factor m, i.e., $\beta_m^n(x) = \beta^n(\frac{x}{m})$. (1) gives

$$\beta_m^n(x) = \frac{1}{m^n}\overbrace{\beta_m^0 * \cdots * \beta_m^0}^{n+1}. \quad (3)$$

$b^n(k)$ represents discrete B-spline satisfying $b^n(k) = \beta^n(k), k \in \mathbf{Z}$. $b_m^n(k)$ means $b^n(k)$ expanded by a factor of m. From $b_m^0(k) = \{\overbrace{1, 1, \cdots, 1}^{m}\}$ we have

$$b_m^n(k) = \frac{1}{m^n}(\overbrace{b_m^0 * b_m^0 * \cdots * b_m^0}^{n+1}) * b^n(k). \quad (4)$$

(4) gives the relation of discrete B-splines between different scales. It says that B-splines can be interpolated by so-called moving averaging operations, and so the functions interpolated by B-spline.

2.2 B-spline Transform

Signals $\{f(k), k \in \mathbf{Z}\}$ interpolated with discrete B-splines b^n are

$$f(k) = \phi^n(k) = \sum_{i=-\infty}^{+\infty} c(i) b^n(k-i). \quad (5)$$

In (5), $\{c(i)\}$ are spline coefficients, $\{f(k)\}$ are represented as the convolution of $\{c(i)\}$ and b^n. The z-transform of (5) is given as

$$F(z) = B^n(z)C(z). \quad (6)$$

Equation (5) gives a result $\phi^n(k)$ by applying a finite impulse response (FIR) operator b^n to $\{c(i)\}$. The spline coefficients $\{c(i)\}$ can be determined simply by inverse filtering, called direct B-spline transform

$$S^n(z) = B^n(z)^{-1} = \frac{1}{\sum b^n(k) z^{-k}}. \quad (7)$$

$S^n(z)$ is a recursive infinite impulse response (IIR) filter. It has been proved to be stable in case n is odd for B-splines[12].

Indirect B-spline transform (6) is used to reconstruct discrete signals from B-spline coefficients. When spline coefficients $\{c(i)\}$ are given, $\{f(k)\}$ can be recovered with (6). Indirect B-spline transform is a finite impulse response (FIR) filter and stable. (7) is called as direct B-spline transform which transform the input signals to spline coefficients.

2.3 Distance transform

In binary images, Distance maps for each pixel equal the distance between that pixel and the pixel of value 1 (in this paper, it means edge pixel) closest to it. There are serveral kinds of distance transform such as Euclidean, city block, chessboard, octagonal and chamfer distances. Euclidean distance map $\mathbf{D} = \{d_{ij}\}$ of a binary image $\mathbf{B} = \{b_{ij}\}$ is defined by

$$d_{ij} = \min_{1 \le p,q \le N} \left\{ \sqrt{(i-p)^2 + (j-q)^2}|_{b_{pq}=0} \right\}. \quad (8)$$

From (8) we can see that Euclidean distance belongs to $\mathbf{L}_2$ distance so it is natural to represent distance

of two pixels in images. In this paper, Euclidean distance is used to calculate distance of pixels to edges for the reason that there is a linear-time algorithm for distance transform[9].

3 The algorithm

The algorithm of image interpolation with edge enhancement is consisted with five steps as follows.

1. The interpolation of images with B-spline β^n. Images expanded for m times both along x and y directions, The expansion result is g_m. By (2), the first derivatives $g'_m|_x$ and $g'_m|_y$ along x and y directions can be derived.

2. Modulus and direction for the first derivative are calculated based on $g'_m|_x$ and $g'_m|_y$. Pixels with local maximum modulus of first derivative (modulus maximum) are regarded as edge points.

3. Edge expansion is performed based on edge pixels. The result is called as Edge Area (EA) with all pixels in it that have the Euclidean distance to edge points no large than $[m/2]$.

4. All outmost pixels of EA form the verge of EA (VEA). Every VEA pixel' intensity is set to be its erode source intensity (ESI). Erosion is performed in EA. In each time of erosion, for each pixel in the neighbor of VEA, compare all neighboring VEA pixels' distance to edges and select one VEA pixel with largest distance as its erode source and copy the ESI to its ESI.

5. For each pixel in EA with intensity V, set its ESI to be E, and distance to edges be d. The edge enhancement is executed

$$V' = E - k(d)(E - V), \tag{9}$$

in (9), $k(d) = \beta^3(2d/m)$.

The coefficient of edge enhancement $k(d)$ get larger to the pixels with large distance. The result of edge enhancement of (9), i.e., new pixel's intensity V', changes smoothly in the result of enhancement.

3.1 B-spline interpolation

Gaussian function is good at describing intensity transient in images. B-spline is a good approximation to Gaussian function. Because there exists a fast algorithm for B-spline interpolation and much calculation is needed to execute the convolution of Gaussian function to images, image expansion is implemented with B-spline in the paper.

With tensor-product[8], we get $\beta_{xy} = \beta_x \beta_y$. 2D B-spline can be calculated with B-spline along x and y-directions. That makes the calculation for 2D B-spline interpolation simple since their bases are separable.

(7) is called the direct spline transform, with which, spline coefficients $C(z)$ are calculated along x and y directions for images.

(5) represents the interpolation of 1D signal $f(x)$. For 2D case, it turns to be

$$f(x, y) = c_{i,j} * b^n(x) * b^n(y), \tag{10}$$

and image expansion is processed as

$$f_m(x, y) = c_{i',j'\uparrow m} * b^n_m(x) * b^n_m(y), \tag{11}$$

where

$$c_{i',j'\uparrow m} = \begin{cases} c_{i,j}, & i' = mi, j' = mj \\ 0, & \text{Otherwise} \end{cases} \tag{12}$$

is the up-sampling of coefficients matrix $c_{i,j}$, while b_m can be calculated by (4), it executes fast by repeating the moving-average[13].

3.2 Edge detection

In Canny edge detection, images are normalized with Gaussian function, and then the first derivative of the smoothed image is calculated. Pixels with modulus maximum are recognized as edge pixels. We find it is time-consuming even as in Mallat's algorithm.

In this paper, edge pixels are detected for edge enhancement. The coefficients $c_{i',j'}$ turned out to be $c'_{i',j'}$ when it is processed by moving average, we get

$$c_{i',j'} * \beta^n_m = c'_{i',j'} * \beta^n \tag{13}$$

about the interpolated images. According to (2), the distribution of first derivative for expanded image along x (or y) direction $g'_m|_x$ (or $g'_m|_y$) is calculated by shifting $c'_{i',j'}$ left (or up) for $m/2$ pixels, right (or down) for $m/2$ pixels and subtracting them. But for $g'_m|_x$ (or $g'_m|_y$),

$$\begin{cases} g'_m|_x = c_{i,j\uparrow m} * b^n_m(y) * b^{n+1}_m(x)', \\ g'_m|_y = c_{i,j\uparrow m} * b^n_m(x) * b^{n+1}_m(y)'. \end{cases} \tag{14}$$

In (14),

$$\begin{cases} b^{n+1}_m(x)' = (b^n_m(x + [m/2]) - b^n_m(x - [m/2])), \\ b^{n+1}_m(y)' = (b^n_m(y + [m/2]) - b^n_m(y - [m/2])). \end{cases}$$

Coefficient matrix should be processed by indirect B-spline transform of order $n+1$ in x (or y) direction and of order n along another direction.

Like previously discussed edge operators, image normalization is also performed in this paper. To avoid the loss of detail information, interpolation is used instead of convolution to normalize images. After interpolation, images have more pixels in it and they turn to be smooth. With $g'_m|_x$ and $g'_m|_y$ derived from (14). the modulus and direction are defined as

$$\begin{cases} \text{Mod} = \sqrt{g'_m|_x^2 + g'_m|_y^2} \\ \theta = \arctan\left(\frac{g'_m|_y}{g'_m|_x}\right) \end{cases} . \qquad (15)$$

In (15), Mod represents the intensity of the first derivative of input image, and θ is the direction along which the value of first derivative changes most sharply near the pixel.

All pixels are recognized as edge pixels when the Mod goes to local maximum value along θ. To speed-up the processing, the searching of maximum Mod is limited to $\theta = 0^{\text{O}}, 45^{\text{O}}, 90^{\text{O}}$ and 135^{O}.

3.3 Erosion

In step four, when each pixel in EA is eroded, much calculation is used to determine the erode source so it is time-consuming. Furthermore, more than one new erode sources will be generated from an erode source in the erosion, this makes the erosion algorithm complicated.

In this paper, a stack is used to execute the erosion. Direction for erosion is determined in the reversed way to what is in erosion.

1. Select an un-eroded EA pixel and push into a stack.

2. Check all neighboring pixels. If there is a VEA pixel or eroded EA pixel, then copy ESI value of the pixel to all elements in the stack, mark them as eroded EA pixels and pop all elements.

3. If there is no VEA pixel or eroded EA pixel, select one neighboring pixel that has the largest distance value and push it in the stack. The search procedure will end when VEA pixel or eroded EA pixel is encountered.

When all EA pixels are eroded, the erosion ends.

In above search algorithm, all erode pixels form chains when they are eroded, with stack, the search algorithm is simplified and executes fast.

4 Experimentals

In this paper, cubic B-spline is used to perform the image interpolation, edge detection, and edge enhancement.

The first test image in this paper is a part of Barbara which is shown in Figure 1. In the image, there are many strips in the scarf of the woman. Before edge detection, image is expanded for $m = 5$ times with linear interpolation. After interpolation, edges in the image are blurred and become widened as shown in Figure 3 (a). Then the derivatives of the image along x and y directions are calculated with (14). All pixels with modulus maximum are regarded as edge points. Distance transform is performed based on edge pixels detected. Since the image is expanded for m times, EA is defined as such area which is composed of pixels with distance equals to or less than $[m/2]$. Through the erosion of EA, erosion intensity for all pixels in EA is calculated. With (9), edge enhancement is performed in EA repeatedly. Images in (b), (c), and (d) of Figure 3 are results for enhancement once, twice, and three times, respectively.

Figure 1: The image is a part of Barbara in which many strips exist in the scarf of the woman.

The second test image in this paper is a part of Lena which is shown in Figure 2. In the image, edges of eye and hat of the woman will be enhenced by the algorithm discussed in the paper. Just the same as the processing upon Figure 1, image is expanded for $m = 5$ times as shown in Figure 4 (a). While (b), (c), and (d) of Figure 4 are results for edge enhancement once, twice, and three times upon (a), respectively.

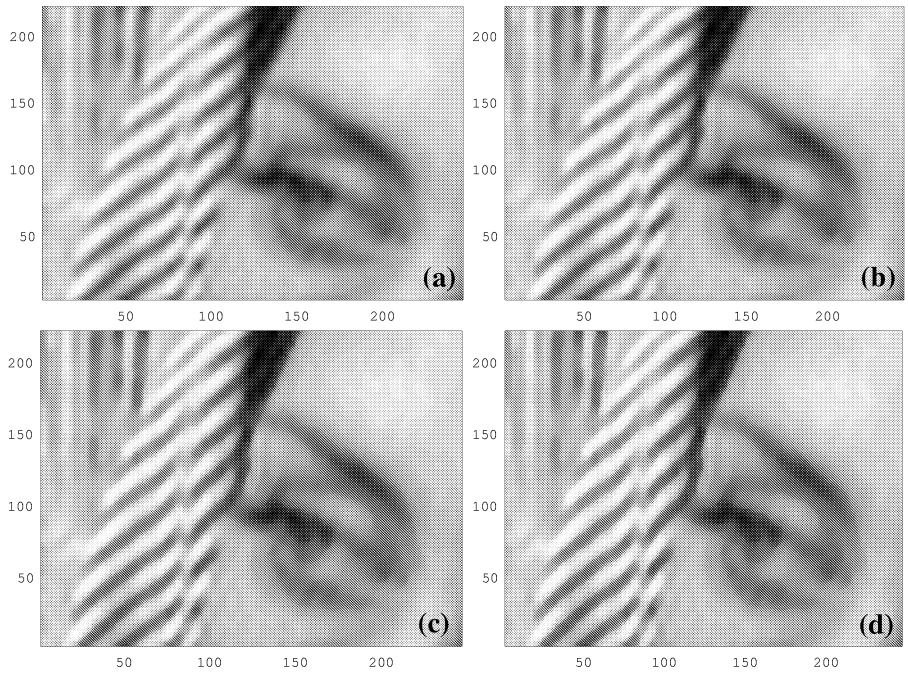

Figure 3: (a) is the result when image is expand r five times; (b) is the result after applying edge enhancement once; (c) is the result for edge enhancement twice; and (d) is the result of edge enhancement for three times.

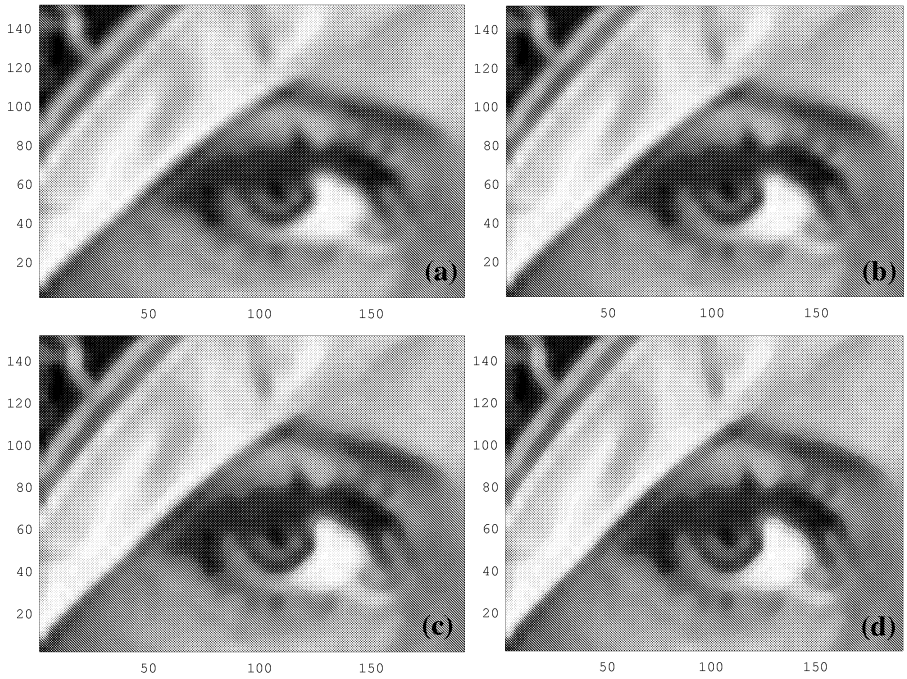

Figure 4: (a) Lena is expanded for five times; (b) Result of edge enhancement once upon Lena; (c) Result of edge enhancement twice upon Lena; and (d) is the result of edge enhancement for three times.

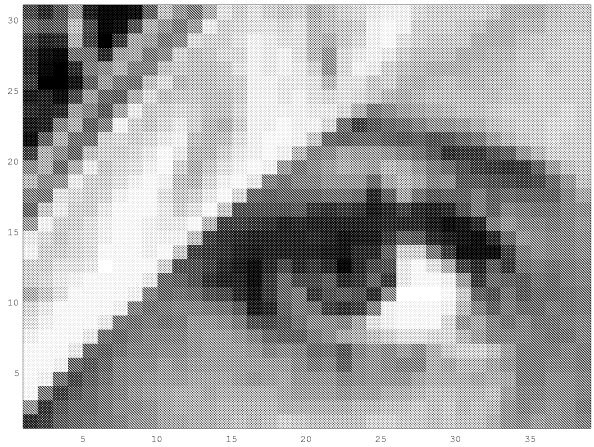

Figure 2: The image is a part of Lena, edges of eye and hat of the woman will be enhenced by the algorithm discussed in the paper.

5 Conclusions

In this paper, an algorithm for Image Interpolation with Edge Enhancement is discussed. To make detail edges in images more perceptible, images are expanded with spline interpolation. Intensity of all newly inserted pixels is determined based on the intensity distribution around it. After interpolation, edges exist in images are detected. Unlike other edge operators that convolute a smoothing function to images directly, the algorithm discussed in this paper smoothes the image by interpolation in which the image is expanded and smoothed, and thus detailed information remains. It executes fast for the reason that it integrates edge detection and interpolation into one algorithm and reduces the edge detection to the operation of subtraction only.

In order to make images smooth even after edge enhancement, the degree of intensity adjustment changes with the Euclidean distance from pixel to edge points. A unified linear-time algorithm for distance transform is used to deal with the calculation of Euclidean distance. EA can be determined when distance is calculated. Erosion is taken place in EA to determine the ESI.

Based upon above results, from (9), edge enhancement can be performed several times to one image. By repeating the edge enhancement, edges in images get clearer.

References

[1] J. Canny, *A Computational Approach to Edge Detection*, *IEEE* Trans. PAMI, vol. 8, no. 6, pp. 679-698, November 1986.

[2] J. J. Clark, *Authenticating Edges Produced by Zero-Crossing Algorithms IEEE* Trans. PAMI, vol. 11, no. 1, pp. 43-57, November 1986.

[3] W. Lawton, S. L. Lee, Z. Shen, *Characterization of compactly supported refinable splines*, Advances in Computational Mathematics, vol. 3, pp. 137-145, 1995.

[4] Stéphone Mallat, *A wavelet Tour of Signal Processing*. Academic Press, 1998.

[5] T. Poggio, H. Voorhees, and A. Yuille, *A Regularized Solution to Edge Detection*, MIT Technical Reports, AIM-833, April 1985.

[6] C. H. Reinsh, *Smoothing by spline functions*, Numer. Math., vol. 10, pp. 177-183, 1967.

[7] I. J. Schoenberg, *Cardinal interpolation and spline functions*, J. Approximation Theory, vol. 2, pp. 167-206, 1969.

[8] L. L. Schumaker, *Spline Functions: Basic Theory*, New York: Wiley, 1981.

[9] T. Hirata, *A Unified linear-time algorithm for computing distance maps*, Information processing letters, 58, pp. 129-133, 1996.

[10] M. Unser, A. Aldroubi, and M. Eden, *B-spline signal processing: Part I-theory*, *IEEE* Trans. Signal Processing, vol. 41, no. 2, pp. 821-833, February 1993.

[11] M. Unser, A. Aldroubi, and M. Eden, *B-spline signal processing: Part II-efficient design and applications*, *IEEE* Trans. Signal Processing, vol. 41, no. 2, pp. 834-848, February 1993.

[12] M. Unser, A. Aldroubi, and M. Eden, *Fast B-spline transforms for continuous image representation and interpolation*, *IEEE* Trans. PAMI, vol. 13, no. 3, pp. 277-285, March 1991.

[13] Yu-Ping Wang, *Image representations using multiscale differential operators*, *IEEE* Trans. IP. Final version, March. 1999.