MVA '96 IAPR Workshop on Machine Vision Applications, November. 12-14, 1996, Tokyo, Japan

A Study on Subpixel Smoothing for Binary Images

Megumi Kawata*
Clarion Co., Ltd.

Jack Koplowitz †
Department of Electrical &
Computer Engineering
Clarkson University

Masato Sakata ‡
Department of Electrical &
Electronic Engineering
Akita University

Abstract

Smoothing of the digitized object boundary or subpixel reconstruction of the original image is necessary in processing digitized images and in computer graphics display of images.

We introduce a reconstruction algorithm by polygonal approximation for curving edges proposed in [1] to estimate the digitized object boundary. This method gives a good subpixel estimate for the object boundary. However, for curving boundaries it does not necessarily give a smooth or visually pleasing result. Therefore, we propose an expanded reconstruction algorithm using B-splines. Experimental results show significant subpixel reconstruction.

1. Introduction

Digital binary images that consist of continuous straight or curving line boundaries are used in diverse fields such as fax machines, digital copy machines, and computers. Digital binary images are obtained using a grid of pixels or square cells, with side T, which are colored either black or white. The pixel level is determined by whether the center point of the cell lies in the black or white region. The resulting digitized silhouette consists of regions whose boundaries are links of horizontal and vertical line segments of pixel length T. Then, the digitized boundary can be represented as a 4-directional chain code with links whose end points are not at the center of a pixel cell but at one of its corners. The link end points are called chain points which form a discrete representation of the boundary curve. Reconstructing the image then consists of fitting a curve near to the chain points.

Such reconstruction can be done in a number of ways. Here, we consider using an approximation for which both the original and the reconstruction give the same digital straight line in the local region around each chain point. A chain code sequence is said to be a digital straight line if there exist straight lines with that representation.

In section 2, we introduce a reconstruction algorithm by polygonal approximation for curving edges. Then in section 3, we give an expanded reconstruction algorithm using the B-spline which gives visually pleasing result for use with curving edges. Then experimental results are presented.

2. Reconstruction Algorithm

Suppose a straight line is digitized and a digitized boundary is obtained as shown in Fig. 1. Let denote a coordinate of a chain point is given by (i, j), i=1,2,3,..., j=1,2,3,.... If (i, j) and $(i+1, j)$ are adjacent chain points, then the pixel center point $(i+1/2, j+1/2)$ must lie above the boundary and the pixel center point $(i+1/2, j-1/2)$ must lie below the boundary. In Fig. 1, the center points lying above the boundary are shown as squares, and below the boundary are shown as circles. Then, we find the squares form the upper convex hull and circles form the lower convex hull. Let us denote them as A_c and B_c, respectively and let l_m be the minimum length line joining A_c and B_c. The algorithm then causes the reconstruction line to be the perpendicular bisector of the line l_m [2].

For curving boundaries, a modified procedure is proposed. From the chain point, extend the chain sequence in both directions an equal amount. The chain sequence is extended as far as possible so that the chain sequence is still a digital straight line. From this chain sequence we can obtain a particular "reconstructed" line which satisfies the chain sequence constraint, i.e. its digitization gives the same chain sequence. The smoothed or reconstructed point is then the original chain point projected onto the reconstructed line [1]. The polygon formed by these projected points form the

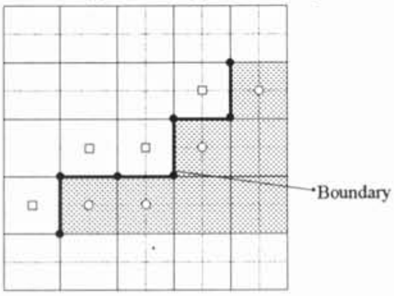


Fig. 1. Links for chain code sequence 100101.

*Address: 50 Kami-toda, Toda, Saitama 335 Japan
E-mail: JBB01207@niftyserve.or.jp
† Address: Clarkson Univ. Box 5720, Potsdam, NY 13699 U.S.A.
E-mail: koplowitz@sun.soe.clarkson.edu
‡ Address: 1-1 Gakuen-cho Tegata, Akita 010 Japan
E-mail: sakata@quartet.akita-u.ac.jp

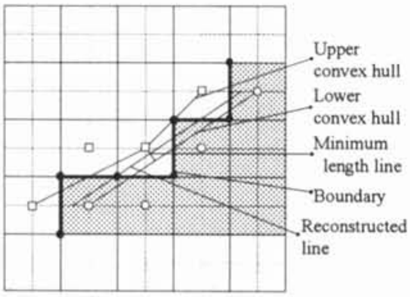


Fig. 2.(a) Constraints and reconstructed line.

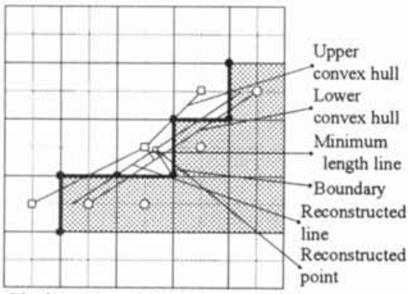


Fig. 2.(b) Chain point shifted by (-.308, .462) gives projection onto reconstructed line.

Table 1. Coordinates of projected points.

CHAIN CODE	SHIFT	CHAIN CODE	SHIFT
000000	(0,0)	0000	(0,0)
000001	(-.019,.096)	0001	(-.05,.15)
000010	(-.059,.235)	0010	(-.2,.4)
000100	(-.15,.45)	0100	(.2,-.4)
001000	(.15,-.45)	1000	(.05,-.15)
001001	(.2,-.4)	0101	(.25,-.25)
001010	(.154,-.231)	1010	(-.25,.25)
010000	(.059,-.235)	1001	(0,0)
010001	(.05,-.15)	01	(-.25,.25)
010010	(0,0)	10	(.25,-.25)
010100	(-.154,.231)	00	(0,0)
010101	(-.25,.25)		
100000	(.019,-.096)		
100001	(0,0)		
100010	(-.05,.15)		
100100	(-.2,.4)		
100101	(-.308,.462)		
101001	(.308,-.462)		
101010	(.25,-.25)		

reconstructed boundary. This procedure is illustrated in Fig. 2 for a chain point with the three links on either side forming the chain code sequence 100101. For fast implementation, a look up table is stored as to all chain link sequences in the first octant of even length up to six along with the corresponding shift of the projected point from the chain point, which is shown in Table 1.

3. Reconstruction Algorithm by the B-spline

It is possible to use a polygonal approximation to estimate curving object, but the reconstructed object boundary would be a set of straight lines. Here, we expand the reconstruction algorithm to curving objects by using a B-spline. In the new algorithm, first, we apply the reconstruction algorithm by polygonal approximation to a digitized object and obtain projected points. Then, we apply B-spline smoothing to the projected points to give the estimation of curving objects. Generally, a B-spline curve is represented as a multinomial function, but we can consider weights for the projected points by using a rational function for a B-spline curve. A variety of curving lines can be expressed by considering weights for each projected point using a rational B-spline curve. Let us denote $P_0, P_1, \ldots, P_n$ as control points (projected points here), and positive real numbers $W_0, W_1, \ldots, W_n$ as weights for each control point, respectively. Let q be a positive number and $\phi(q) = \lfloor q/2 \rfloor$, where $\lfloor x \rfloor$ indicates the biggest integer less than x. Using the B-spline function $N^q(t)$, let us define

$$Q(t) = \frac{\sum_{i=-2\phi(q)}^{n+2\phi(q)} W_i N^q(t-i) P_i}{\sum_{j=-2\phi(q)}^{n+2\phi(q)} W_j N^q(t-j)} \qquad (-\phi(q) \le t \le n + \phi(q)) \tag{1}$$

Equation (1) $\{Q(t) | 0 \le t \le n\}$ is called qth power of the rational B-spline curve with control points $P_0, P_1, \ldots, P_n$ and weights $W_0, W_1, \ldots, W_n$ [3].

Here, the projected points are estimated by any of two, four or six links of chain code sequences. Therefore, we define three types of weights, w_2, w_4 and w_6 for each case of the projected points. We have to consider two issues with respect to the weights: one is the accuracy of projected points and the other is to satisfy the chain code constraint on the reconstructed image. First, each type of the projected point contains different accuracy. Let us call the points estimated by six links of chain code sequences as type (i), estimated by four links of chain code sequences as type (ii), and estimated by two links of chain code sequences as type (iii), respectively. Then, the accuracy of type (ii) points should be greater than that of type (iii) points. Similarly, the accuracy of type (i) points should be greater than that of type (ii) points. It gives the first condition on weights as follows: weights have to be greater in order of importance of the points. Assuming the weight in type (iii) w_2=1.0, the weight in type (ii) w_4 and the weight in type (i) w_6 can be expressed as $1 \leqq w_4 \leqq w_6$. Second, digital binary images are obtained using a grid of pixels or square cells which are colored either black or white depending on whether the center point of the cell lies in the black or white region. If a

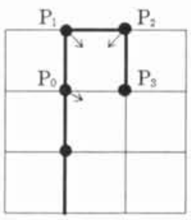


Fig. 3. The worst case of a chain code sequence.

reconstructed point is located over the center point of the pixel, the redigitized image would be different from the original digitized image. Fig. 3 shows the worst case of a chain code sequence for which it is not possible to satisfy the chain code constraint on the reconstructed image. Consider the case in Fig. 3 and apply a cubic B-spline function to reconstruct P_1. Assuming $P_0(0, 0)$, $P_1(0, 1)$, $P_2(1, 1)$ and $P_3(1, 0)$, the following equation has to be satisfied.

$$P_{1y} = \frac{w_0 N^3(t+1) P_{0y} + w_1 N^3(t) P_{1y} + w_2 N^3(t-1) P_{2y} + w_3 N^3(t-2) P_{3y}}{w_0 N^3(t+1) + w_1 N^3(t) + w_2 N^3(t-1) + w_3 N^3(t-2)} > 0.5 \tag{2}$$

By calculating (2), the second condition $w_4 < 2.2727...$ is obtained. Overall, weights have to be decided as $w_2 = 1.0$, $1.0 \leqq w_4 < 2.27$, $w_4 \leqq w_6$.

4. Experimental Methods

A circle is considered as a good example of a curving object, because a circle can express various curvatures by varying the radius. Hence, we use digitized circles with various radiuses to decide weights of the reconstruction algorithm by the B-spline. In the followings, we show the methods and the procedures of experiments.

Experiment 1

Object: Circles (center points are located on the grid points.)

Procedure:

1) Define the normalized average error e of reconstructed points for the average radius of a reconstructed circle as follows:

$$e = \sqrt{\frac{\sum_{i=1}^{n} (r_i - r_{ave})^2}{n-1}} \Bigg/ r_{ave} \tag{3}$$

and define the weights that minimize e as optimized weights. For each pattern of circle, optimize the weights, w_4 and w_6.

2) From the weights obtained in 1), set four combinations of weights, w_4 and w_6 including the case of $w_4 = w_6 = 1$ for comparison. Calculate e for each combination of weights for each pattern of circle.

3) From the result in 2), discuss the recommended combination of weights.

Experiment 2

Object: Circles (center points are not located on the grid points.)

Procedure:

1) Calculate e for four combinations of weights set in experiment 1.

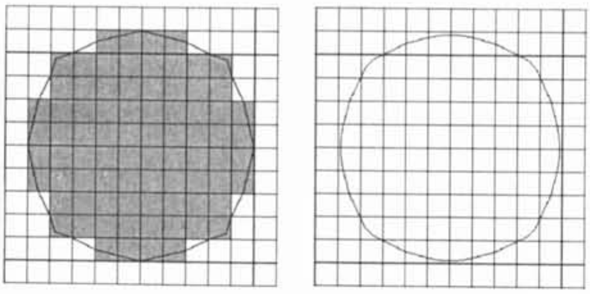
2) From the result in 1), discuss the recommended combination of weights.

5. Experimental results

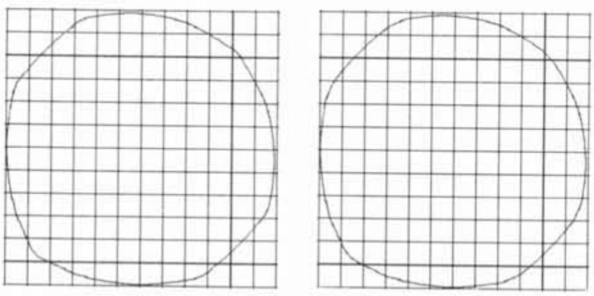
Table 2 shows the results when optimizing the weights so that the normalized error e gives the minimum value. The radius varies between 32 and 220 for pixel length 32. Eighteen patterns of circles are shown and the average radius of the digitized circles with the optimized weights varies between 29.751611 and 215.428620. In pattern 1, since the radius versus pixel length is less than 1, the projected points are estimated only by two links of chain code sequences. Therefore, w_4 and w_6 can take any values. In pattern 2 and 3, the radius versus pixel length is still small and take the value between 1.77 and 2.07. In this case, the projected points are estimated by either two links of chain code sequences or four links of chain code sequences. Therefore, w_6 can take any value. In pattern 5 and 6, w_6 takes close value to w_4 such as 3.1 or 2.3 for $w_4 = 2.2$. In this case, the radius versus pixel length varies between 3.01 and 3.66. After pattern 7, w_6 takes much greater value than w_4 except for pattern 11. + in table 2 shows that the weights can be decided only by w_4 and w_6, since there is no projected point estimated by two links of chain code sequence. As the radius gets bigger, there are some cases that w_2 and w_4 can take any value, since all the projected points are estimated by six links of chain code sequences. * means the normalized error e is calculated as $w_6 = 25$.

Table 2. Experimental results.

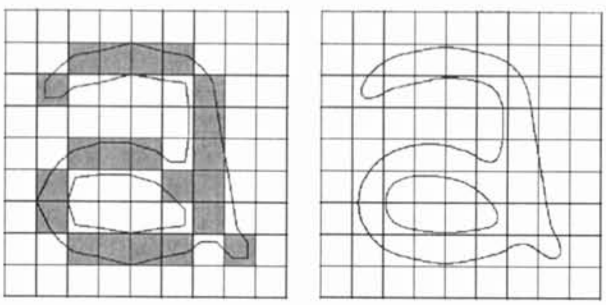
Pattern	r_{ave}	w_4	w_6	e	w(1.1.1)	w(1.2.3)	w(1.2.4)	w(1.2.6)
1 (32.0)	29.751611	any	any	0.015030	0.015030	0.015030	0.015030	0.015030
2 (64.0)	56.574631	2.2	any	0.009565	0.026900	0.011554	0.011554	0.011554
3 (80.0)	66.129064	2.2	any	0.047630	0.063888	0.049745	0.049745	0.049745
4 (88.0)	81.903658	2.2	20.7	0.044968	0.056645	0.048451	0.047994	0.047263
5 (96.0)	96.397577	2.2+	3.1	0.033256	0.033320	0.033257	0.033335	0.033712
6(120.0)	117.062620	2.2	2.3	0.018240	0.024810	0.019139	0.019242	0.019414
7(128.0)	124.084246	2.2+	8.0	0.003898	0.008499	0.006610	0.005313	0.004058
8(144.0)	134.266983	2.2+	infinity*	0.045804	0.054286	0.052979	0.051890	0.050233
9(148.0)	144.841492	2.2	infinity*	0.023683	0.029095	0.024899	0.024800	0.024629
10(156.0)	151.671863	2.2+	infinity*	0.018780	0.024133	0.022980	0.022146	0.021044
11(160.0)	157.701192	2.2	3.8	0.015423	0.019551	0.015914	0.015910	0.016008
12(176.0)	164.452231	2.2+	infinity*	0.023087	0.026624	0.025806	0.025237	0.024509
13(182.0)	173.402147	2.2	infinity*	0.029747	0.036190	0.032679	0.032288	0.031715
14(192.0)	186.494769	2.2+	20.6	0.002249	0.007022	0.005835	0.004966	0.003815
15(200.0)	193.275312	2.2+	infinity*	0.007545	0.012426	0.011362	0.010609	0.009619
16(208.0)	197.740047	2.2	infinity*	0.026920	0.031704	0.029062	0.028739	0.028284
17(212.0)	211.062358	2.2	19.5	0.013264	0.015875	0.014021	0.013885	0.013707
18(220.0)	215.428620	2.2+	infinity*	0.009041	0.011760	0.011094	0.010650	0.010094



(a)Polygonal approximation. (b)B-spline with weights(1,2,4).
Fig. 4. Reconstructed circles (*Experiment 1*, Pattern 11).



(a)B-spline with weights(1,1,1). (b)B-spline with weights(1,2,6).
Fig. 5. Reconstructed circles (*Experiment 2*).



(a)Polygonal approximation. (b)B-spline with weights(1,2,3).
Fig. 6. Reconstructed "a"s.

We set four combinations of weights depending on the result in experiment 1 as follows: i) from pattern 5 and 6, w_2=1. w_4=2, and w_6=3, ii) from pattern 11, w_2=1, w_4=2, w_6=4, iii) from patterns after pattern 7, w_2=1, w_4=2, w_6=6, and iv) w_2=w_4=w_6=1. The normalized errors with these four patterns are shown on the right half of table 2. This results show that $w(1, 2, 3)$ causes the minimum normalized error up to pattern 6, and $w(1, 2, 6)$ causes the minimum normalized error in the most of the patterns after pattern 7. Fig. 4 shows circles reconstructed by polygonal approximation and the B-spline with $w(1, 2, 4)$. Fig. 5 shows the result of experiment 2. The center point of the circle is moved to (28, -6) for pixel length 32. Fig. 6 shows the result when the algorithm is applied to general images.

6. Conclusions

By using a polygonal reconstruction algorithm, fast and good quality reconstruction of digital binary images has been achieved. In this study, high quality reconstruction of curving images is achieved using a reconstruction algorithm by the B-spline. In the new algorithm, we apply B-spline smoothing for projected points. By using rational B-spline curves, we can consider the weights for each projected point. The projected points are estimated by any of two links of chain code sequences, four links of chain code sequences, or six links of chain code sequences. Therefore, we define three types of weights, w_2, w_4, and w_6 for each case of the projected points. Here, we have to consider two issues with respect to weights: one is the accuracy of projected points and the other is to satisfy the chain code constraint on the reconstructed image. Assuming w_2=1.0, it is necessary that $1.0 \leqq w_4 < 2.27$, $w_4 \leqq w_6$. From experimental results, recommended weights for curving objects are decided. Table 3 shows the recommended combinations of weights as to radius versus pixel length. As the radius gets larger, reconstructed points estimated by six links of chain code sequences are increasing. In those cases, smoother curves are obtained by strengthening the weight w_6.

General images consist of both curving and straight line boundaries and curves have various curvatures: therefore, small weights are appropriate. Experimental results show that the combination of weights, w_2=1.0, w_4=2.0, and w_6=3.0 is appropriate for recommended weights for general curving objects.

Table 3. Recommended combinations of weights.

radius versus pixel length	w_2	w_4	w_6
less than 4	1	2	3
greater than 4	1	2	6

References

[1] J.Koplowitz, "Scaling of Digital Shapes with Subpixel Boundary Estimation", SPIE Vol. 1470, Data Structures and Target Classification, pp. 167-174, 1991.

[2] J. Koplowitz and A. P. Sundar Raj, "A Robust Filtering Algorithm for Subpixel Reconstruction of Chain Coded Line Drawings", IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. PAMI-9, pp. 451-457, May 1987.

[3] Atsuyoshi Sugihara, "Mathematics for graphics", Kyouritsu Press, 1995.