Journal of Zhejiang University-SCIENCE C (Computers & Electronics)
ISSN 1869-1951 (Print); ISSN 1869-196X (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

Curve length estimation based on cubic spline interpolation in gray-scale images*

Zhen-xin WANG[1,2], Ji-hong OUYANG[‡1,2]
([1]College of Computer Science and Technology, Jilin University, Changchun 130012, China)
([2]MOE Key Laboratory of Symbolic Computation and Knowledge Engineering, Jilin University, Changchun 130012, China)
E-mail: wzhx_cc07@mails.jlu.edu.cn; ouyj@jlu.edu.cn

Received Feb. 28, 2013; Revision accepted May 6, 2013; Crosschecked Sept. 16, 2013

Abstract: This paper deals with a novel local arc length estimator for curves in gray-scale images. The method first estimates a cubic spline curve fit for the boundary points using the gray-level information of the nearby pixels, and then computes the sum of the spline segments' lengths. In this model, the second derivatives and *y* coordinates at the knots are required in the computation; the spline polynomial coefficients need not be computed explicitly. We provide the algorithm pseudo code for estimation and preprocessing, both taking linear time. Implementation shows that the proposed model gains a smaller relative error than other state-of-the-art methods.

Key words: Arc length estimation, Cubic spline interpolation, Gray-scale image, Local algorithm
doi:10.1631/jzus.C1300056 **Document code:** A **CLC number:** TP751

1 Introduction

The length computation of a continuous curve segment in a digital image is a basic problem in numerous applications, such as feature analysis (Deng *et al.*, 2006; Zhang *et al.*, 2011) and distance estimation in the real or virtual world (Tadrous, 2010). At first glance, the estimation process may be thought of as simple, but, in fact, it is a challenging task to obtain an accurate result. This is due to the partial loss of digital image boundary information in the digitization step. Many researchers have begun to focus on how to compute the continuous curve segment length (Coeurjolly and Klette, 2004; Sladoje and Lindblad, 2009; Cai and Walker, 2010; Suhadolnik *et al.*, 2010) since the Freeman chain code (Freeman, 1961) was proposed.

Most of the algorithms of the contour length estimator in binary images (Cai and Walker, 2010) are based on local metrics, polygonalizations of digital curves (maximum-length digital straight segments, i.e., DSSs), or minimum length polygons (MLPs). Recent work by Lachaud and Provençal (2011) improved the traditional estimator at the execution level. However, methods dealing with binary images do not take into consideration the sub-pixel information. We note a gradient-based method for preciseness (Coeurjolly and Klette, 2004), spline interpolation methods to calculate the ground relief contours (Foteinopoulos, 2009), and calculation of curve length based on B-spline (Suhadolnik *et al.*, 2010). The advantage of continuity and smoothness in curve approximation has been shown (Foteinopoulos, 2009; Suhadolnik *et al.*, 2010).

Sladoje and Lindblad (2009) addressed the estimation accuracy problem by taking into account the gray levels of the pixels at the boundary after digitization. The basic assumption is that the pixel intensity

‡ Corresponding author
* Project supported by the National Natural Science Foundation of China (Nos. 61170092, 61133011, 61272208, 61103091, and 61202308) and the Fundamental Research Funds for the Central Universities, China (Nos. 450060445674 and 450060481512)

is proportional to the area of the pixel covered by the object's silhouette. The algorithm manages to calculate the local length of the straight line in a 3×3 region with the central pixel located at the object boundary.

To make length estimation more accurate and precise, in this work there is first an estimate of a cubic spline curve fit for the boundary points using the gray-level information, which is according to pixel coverage digitization (Sladoje and Lindblad, 2009). We then calculate the length of the cubic spline. For computational efficiency, we consider only the first two items of the Taylor polynomials at the given point. Note that this work is different from Foteinopoulos (2009), who used precise coordinates, instead of digital images, as input. The main difference between this work and Suhadolnik *et al.* (2010) is that we employ cubic spline interpolation and a local estimation algorithm with 4×*N* configuration (Fig. 1), while they used the approximation with a B-spline.

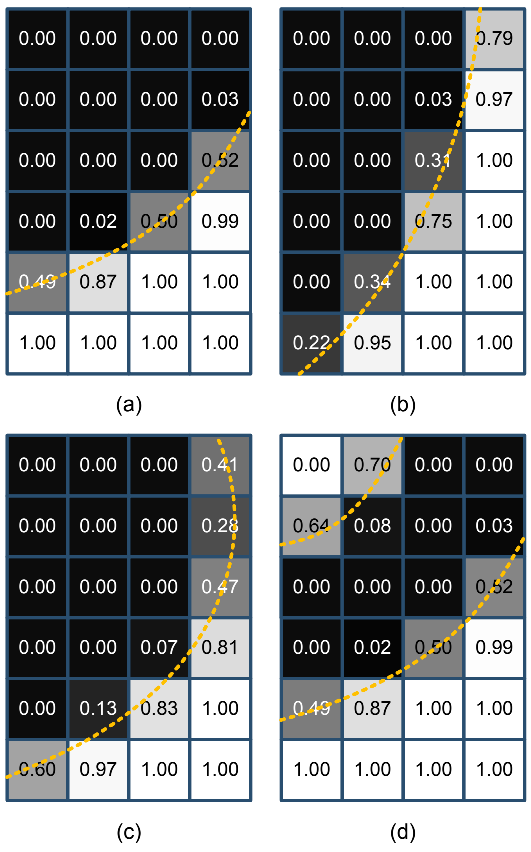

Fig. 1 Edge length estimation based on cubic spline interpolation with 4×*N* configuration
(a) Normal case; (b) Large slope; (c) Vertical tangent line; (d) Two edges are too close

2 Pixel coverage digitization

Like Sladoje and Lindblad (2009), in this work we assume that there exists a segmentation method that approximately provides pixel coverage digitization. Pixel coverage digitization means that the gray level of each pixel in the image is the ratio of the foreground area to the area of the corresponding pixel. The formal definitions of pixel coverage digitization and *n*-level quantized pixel coverage digitization are given as follows (Sladoje and Lindblad, 2009):

Definition 1 (Pixel coverage digitization) For a given continuous object $S\in\mathbb{R}^2$, inscribed into an integer grid with pixels $p_{(i,j)}$, the pixel coverage digitization of S is

$$D(S)=\left\{\left((i,j),\frac{A(p_{(i,j)}\cap S)}{A(p_{(i,j)})}\right)\middle|\,(i,j)\in\mathbb{Z}^2\right\},$$

where $A(X)$ denotes the area of a set X.

Definition 2 (*n*-Level quantized pixel coverage digitization) For a given continuous object $S\in\mathbb{R}^2$, inscribed into an integer grid with pixels $p_{(i,j)}$, the *n*-level quantized pixel coverage digitization of S is

$$D_n(S)=\left\{\left((i,j),\frac{1}{n}\left\lfloor n\frac{A(p_{(i,j)}\cap S)}{A(p_{(i,j)})}\right\rfloor\right)\middle|\,(i,j)\in\mathbb{Z}^2\right\},$$

where $\lfloor x\rfloor$ denotes the largest integer not greater than x. $\lim\limits_{n\to\infty}D_n(S)=D(S)$.

For length estimation based on pixel coverage digitization, it is assumed that the boundary of an object is locally smooth. However, from the point of view of fractal, of which the main idea is a detailed pattern repeating itself, the actual length of the curve segment can be computed as the estimated length multiplied by a constant larger than one. Thus, we focus on length estimation of the locally smooth curve.

3 Length estimation based on cubic spline interpolation in a gray-scale image

3.1 Cubic spline interpolation

The well-known theory of cubic spline interpolation is simple (Dyer and Dyer, 2001). The fundamental idea is to fit a piecewise function s over n intervals from a set of n+1 points $\{(x_i, y_i)|i=1, 2, \ldots, n+1\}$, where $x_i<x_{i+1}$ for i=1, 2, …, n.

$$s(x)=\begin{cases} s_1(x), & x_1 \le x < x_2, \\ s_2(x), & x_2 \le x < x_3, \\ & \vdots \\ s_n(x), & x_n \le x \le x_{n+1}, \end{cases}$$

where s_i (i=1, 2, …, n) is a third-degree polynomial of the form

$$s_i(x)=a_i(x-x_i)^3+b_i(x-x_i)^2+c_i(x-x_i)+d_i. \quad (1)$$

For i=1, 2, …, n,

$$\begin{cases} s_i(x_i)=y_i \\ s_i(x_{i+1})=y_{i+1} \end{cases}$$

results in end-to-end connection between the spline segments. To ensure smoothness, the fact that the cubic spline is of class C^2 means the first and second derivatives of function s at each interior node both exist and are continuous, i.e.,

$$\begin{cases} s_i'(x_{i+1})=s_{i+1}'(x_{i+1}), \\ s_i''(x_{i+1})=s_{i+1}''(x_{i+1}). \end{cases}$$

Since the interval in the application scenario is unit-wide (Fig. 1), let $h=x_{i+1}-x_i$. For simplicity, all the coefficients in Eq. (1) can be rewritten with y_i and y_i'' as

$$\begin{cases} a_i=(y_{i+1}''-y_i'')/(6h), \\ b_i=y_i''/2, \\ c_i=\dfrac{y_{i+1}-y_i}{h}-\dfrac{h(y_{i+1}''+2y_i'')}{6}, \\ d_i=y_i. \end{cases} \quad (2)$$

3.2 Length estimation on the unit-wide interval

Applying the arc length formula

$$l=\int_a^b \sqrt{1+[f'(x)]^2}\,\mathrm{d}x \quad (3)$$

of function $y=f(x)$ over the interval $[a, b]$ to $s_i(x)$, we have

$$l_i=\int_{x_i}^{x_{i+1}} \sqrt{1+[3a_i(x-x_i)^2+2b_i(x-x_i)+c_i]^2}\,\mathrm{d}x. \quad (4)$$

Thus, $t=x-x_i$, and Eq. (4) can be rewritten using the linear approximation expansion of the Taylor series at point $x=x_i$ as

$$l_i(t)=t\sqrt{1+c_i^2}+o(t^2). \quad (5)$$

Eqs. (1)–(5) lead to

$$l_i \approx h\sqrt{1+c_i^2}=h\sqrt{1+\left[\frac{y_{i+1}-y_i}{h}-\frac{h(y_{i+1}''+2y_i'')}{6}\right]^2}. \quad (6)$$

Note that when $s_i(x)$ is a function of line, the right-hand side of Eq. (6) reduces to $h\sqrt{1+[(y_{i+1}-y_i)/h]^2}$, which is the result of length estimation of the line segment.

The basic application scenario of this work is to estimate the arc length on an 8-bit bitmap, which allows 256 different intensities to be recorded, typically on a linear scale.

Assume that the curve is locally smooth and that the image has been pixel coverage digitized. Therefore, we simply estimate the derivative of y with respect to $(x_i+x_{i+1})/2$ by difference of Newton's interpolation polynomial:

$$\widehat{y_{i+0.5}'}=\frac{y_{i+1}-y_i}{h}, \quad (7)$$

where the subscript 'i+0.5' denotes the middle position of i and i+1. The second derivative can be derived from Eq. (7):

$$\begin{aligned} \widehat{y_i''}&=\frac{\widehat{y_{i+0.5}'}-\widehat{y_{i-0.5}'}}{h}=\frac{(y_{i+1}-y_i)-(y_i-y_{i-1})}{h^2} \\ &=\frac{y_{i+1}+y_{i-1}-2y_i}{h^2}. \end{aligned} \quad (8)$$

Eqs. (6)–(8) and h=1 lead to

$$\hat{l}_i=\sqrt{1+\left(\frac{-2y_{i-1}-3y_i+6y_{i+1}-y_{i+2}}{6}\right)^2}, \quad (9)$$

which takes linear time. Now, using Eq. (9), we can estimate the local arc length without explicitly calculating the coefficients of the cubic spline equation.

Note that Eq. (6) and other second derivative estimation methods should be used, if the curve is not locally smooth. Eq. (9) means that the proposed method estimates the arc length as a line segment with slope $\hat{k}=(6y_{i+1}-3y_i-2y_{i-1}-y_{i+2})/6$. Then Eq. (9) can be simplified as follows:

$$\hat{l}_i=\sqrt{1+\hat{k}^2}. \tag{10}$$

3.3 Error analysis

Eq. (4) can be rewritten as $l=\int_{x_i}^{x_i+t}\sqrt{1+[f'(x)]^2}\,\mathrm{d}x$; its second-order Lagrange form of the remainder term is $R(t)=\dfrac{f'(\xi)f''(\xi)t^2}{2\sqrt{1+[f'(\xi)]^2}}$, where ξ is between 0 and t. Obviously, $\lim\limits_{t\to 0}R(t)=0$, i.e., $\lim\limits_{r\to\infty}R(t)=0$, where r is the grid resolution (note that $t=h=1/r$).

Let $q=\{q_j=j/n\,|\,j=0,1,\ldots,n-1\}$ be the n levels of quantized pixel coverage digitization. For each y_i, the maximum error of its quantized value is $\pm\dfrac{1}{2n}$. $\widehat{l_i^{\mathrm{q}}}=\sqrt{1+\left(\hat{k}\pm\dfrac{1}{3n}\right)^2}$. The relative error of length estimate $\hat{l}_i$ with n levels of quantized pixel coverage digitization is given by

$$\varepsilon_n=\frac{\widehat{l^{\mathrm{q}}}-\hat{l}}{\hat{l}}=\sqrt{\left[1+\left(\hat{k}\pm\frac{1}{3n}\right)^2\right]\Big/\left(1+\hat{k}^2\right)}-1. \tag{11}$$

The plot of ε_n (Fig. 3) gives a visual impression of the error function. Usually, the four errors of y values do not reach $\pm\dfrac{1}{2n}$ at the same time, and therefore ε_n is only an upper bound of the relative error. The theoretical maximum error ε is 8.680% (Fig. 3a) in case of pixel digitalization level 2; however, the maximum error in the test results is 3.645% (Fig. 4).

4 Computation of local length

4.1 Three constraints on local estimation

Like Sladoje and Lindblad (2009), we use sums of (quantized) pixel values in columns to estimate the y coordinate. From Eq. (9), we know that the computation needs four neighboring columns of pixels as input. All pixels involved in local calculation constitute the active area (Fig. 2c), which is then turned into the computation area (Fig. 2d). The core problems are to find out how many rows we need in the computation and to determine whether the derivative exists at the estimated position. If the slope of the tangent exceeds a certain threshold, which means the slope is too large or even does not exist, then the row-wise, instead of column-wise, calculation of sums should be made. In case of continuous curve segments, the preprocessing is a bit more complicated.

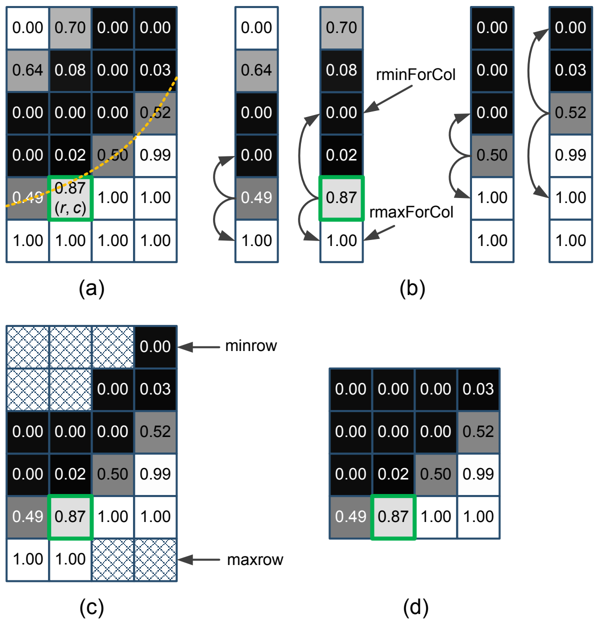

Fig. 2 Example process flow for the proposed algorithm of arc length estimation by cubic spline (Algorithm 1)
(a) Input image with estimation position (r, c); (b) Finding the active area in each column; (c) The whole active area for 4×N configuration; (d) The last data matrix

Fig. 1 shows some examples of different situations in computation. In the normal case (Fig. 1a), the absolute value of the curve slope across the four columns is not very large. This results in a small number of rows being needed for calculation. As seen in Fig. 1b, the curve slope in the 4th column is very high. Fig. 1c shows an example having a vertical tangent line in the fourth column, which is not fit for column-wise computation. However, we can make row-wise calculation of forms as shown in Figs. 1b and 1c. There are two edges in Fig. 1d, in which we cannot easily add up the pixels' values within each column until the other edges are eliminated locally in the image.

To sum up, there are three constraints of local estimation: (1) All pixel values are monotonic in each column in the active area; (2) All columns in the active area have the same monotonicity; (3) The number of rows in the active area does not exceed a certain threshold.

4.2 An algorithm for edge length computation

First locate the edge manually or using other edge detection methods. Then test each edge point to determine if it satisfies the three constraints in case of either row-wise or column-wise calculation. Generally, at least one type of them can pass all the three tests, in which case we can calculate the arc length by Algorithm 1. If not, use the algorithm in Sladoje and Lindblad (2009) to compute the arc length.

Algorithm 1 Arc length estimation by cubic spline

```
for all neighbor columns of point ($r$, $c$) (Fig. 2) do
    Find the pixels between rminForCol and
      rmaxForCol (Fig. 2b)
end for
Compute minrow and maxrow (Fig. 2c)
Obtain the local computation area (Fig. 2d)
Compute all $y$ values and calculate len
```

Due to computational symmetry, we can use the average of the 'forward' and 'backward' computation results of l_i from Eq. (9):

$$\hat{l}_i = h\left(\sqrt{1+\hat{k}_{\mathrm{f}}^2} + \sqrt{1+\hat{k}_{\mathrm{b}}^2}\right)\Big/2, \tag{12}$$

where

$$\hat{k}_{\mathrm{f}} = \frac{-2y_{i-1} - 3y_i + 6y_{i+1} - y_{i+2}}{6},$$

$$\hat{k}_{\mathrm{b}} = \frac{-2y_{i+2} - 3y_{i+1} + 6y_i - y_{i-1}}{6}.$$

The 'forward', 'backward', and 'average' estimation results are shown in Section 5. In the normal case (like in Fig. 1a), we can compute the local arc length by Algorithm 1. Here only the pseudo code for column-wise calculation of the arc length is shown. In case of row-wise calculation, the image should first be rotated by 90°. Although the proposed model is not symmetric for column-wise and row-wise computation, it gives good estimation results, as shown later in Section 5. We recommend using row-wise calculation to estimate the arc length when the estimated slope is k>1, as the relative error is smaller when k<1 (Fig. 3).

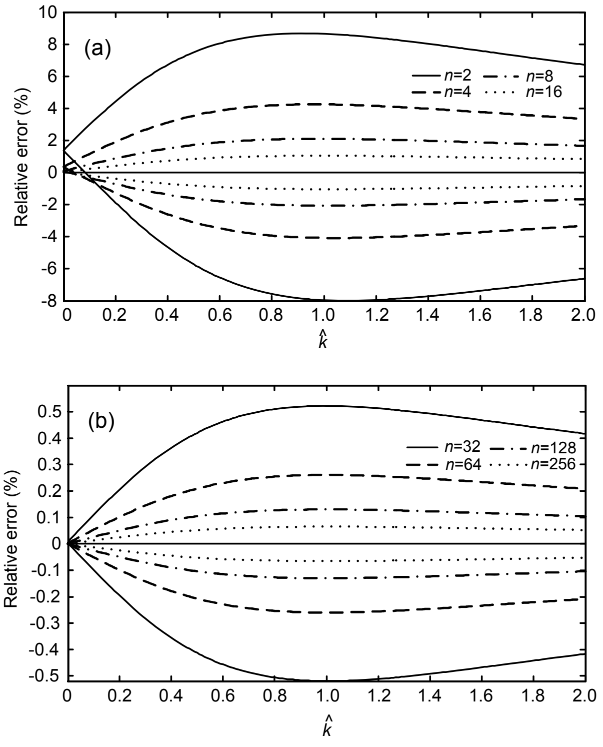

Fig. 3 Relative errors for different quantized pixel coverage digitization levels
(a) n=2, 4, 8, 16, with the maximum relative errors being 8.680%, 4.253%, 2.105%, and 1.047%, respectively; (b) n=32, 64, 128, 256, with the maximum relative errors being 0.522%, 0.261%, 0.130%, and 0.065%, respectively

5 Experiment

5.1 Elementary functions test

We estimate the lengths of eight kinds of arc segments over four resolutions (10, 20, 50, 100) and eight different quantized pixel coverage digitization levels (2, 4, 8, 16, 32, 64, 128, 256), which are composed of some elementary functions. We also test the proposed method on the corresponding Gaussian and average blurred images to simulate real images. In the experiments, there are three filter windows, and the σ of the Gaussian kernel is 0.5. The eight elementary functions and the corresponding estimation intervals are listed in Table 1.

Table 1 Relative errors of the estimated length of curves of different elementary functions[*]

Function	Interval	Relative error (%)		
		Forward	Backward	Average
$y=\sin x$	[0, 2π]	−0.002	−0.017	−0.009
$y=\sqrt{4-x^2}$	[−1, 1]	0.030	0.030	0.030
$y=2^x$	[−2, 0]	−0.430	0.454	0.012
$y=x/2$	[0, 2]	0.000	0.000	0.000
$y=\log_2 x$	[0.5, 1.5]	0.716	−0.661	0.027
$y=x^2/2$	[0, 1]	−1.847	1.952	0.052
$y=x^3/3$	[0, 1]	−1.445	1.709	0.132
$y=x^4/4$	[0, 1]	−1.057	1.380	0.161

[*] Grid resolution=10×10

All relative errors of the three kinds of calculation (forward, backward, and average) at grid resolution 10×10, digitalization level $n=\infty$ are listed in Table 1. Fig. 4 shows the relative errors of the estimation at different quantized pixel coverage digitization levels and with different grid resolutions. Figs. 5 and 6 show the estimation results on the Gaussian and average blurred pixel coverage digitization images, respectively. The test results of estimation on line $y=x/2$ are not presented in these figures, as its errors are relatively small compared with the errors of the other seven curves and adding one more curve will reduce the readability of the figures. It can be observed from Figs. 4–6 that the relative errors converge and fall in a certain range as the grid resolution increases.

5.2 Synthetic images test

We apply the proposed algorithms on a set of synthetic shapes digitized with pixel coverage digitization (Fig. 7), which was presented in Klette *et al.* (1999) and also used by Coeurjolly and Klette (2004), Sladoje and Lindblad (2009), and Suhadolnik *et al.* (2010).

Our test method is almost the same as that used by Sladoje and Lindblad (2009). However, for accuracy, for each test shape and each of the 20 different rotations and positions, we generate the pixel coverage digitization image with resolutions in the range of [10, 1024]. Fig. 8 presents the evaluation results. Although the amplitude of relative errors increases as the grid resolution increases, the local maximum of the error is lower than that of Sladoje and Lindblad (2009)'s algorithm. Unlike Suhadolnik *et al.* (2010),

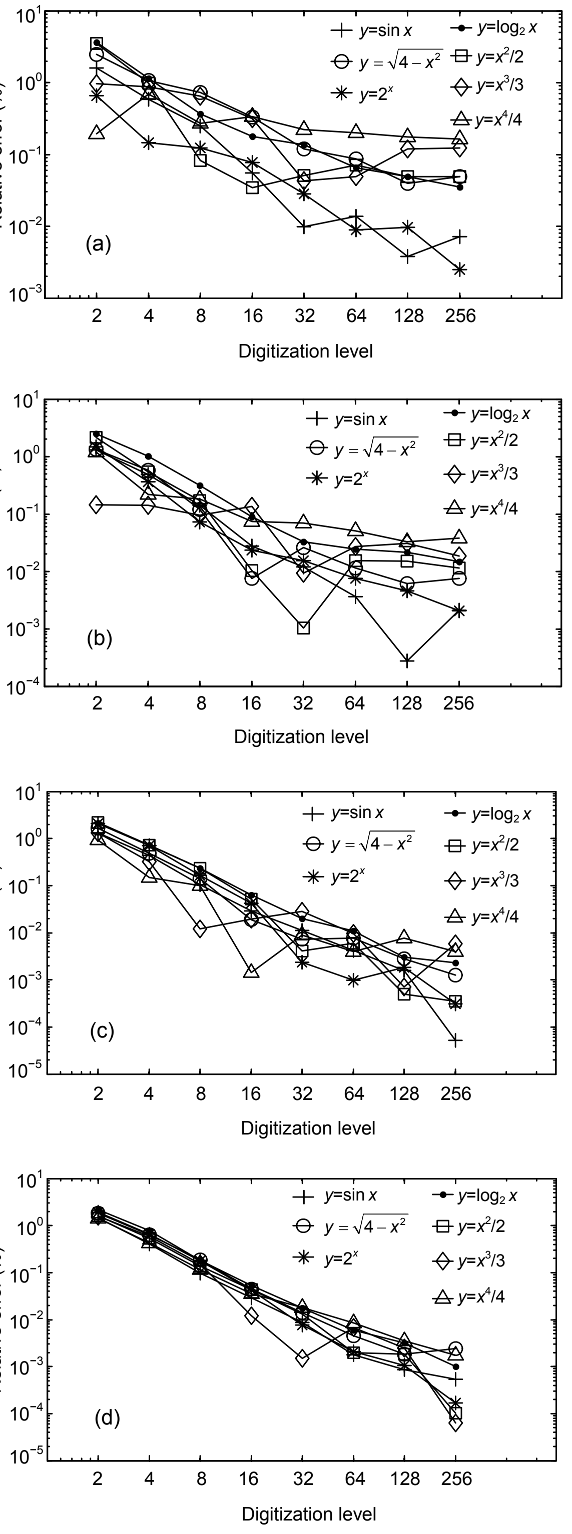

Fig. 4 Relative errors for seven different curve functions
(a) Grid resolution=10, $\varepsilon^{\max}$=3.645%; (b) Grid resolution=20, $\varepsilon^{\max}$=2.505%; (c) Grid resolution=50, $\varepsilon^{\max}$=2.200%; (d) Grid resolution=100, $\varepsilon^{\max}$=2.176%

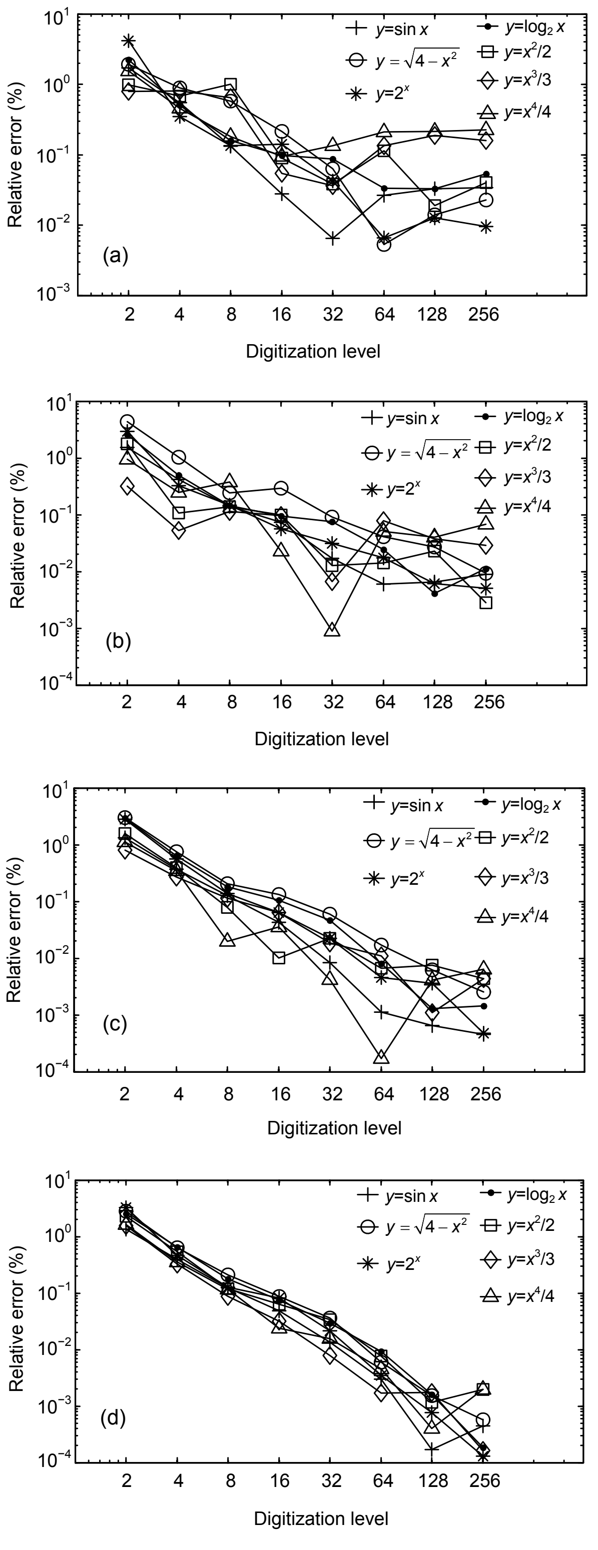

Fig. 5 Relative errors for Gaussian blurred elementary function images

(a) Grid resolution=10, $\varepsilon^{\max}$=4.126%; (b) Grid resolution=20, $\varepsilon^{\max}$=4.330%; (c) Grid resolution=50, $\varepsilon^{\max}$=3.014%; (d) Grid resolution=100, $\varepsilon^{\max}$=3.235%

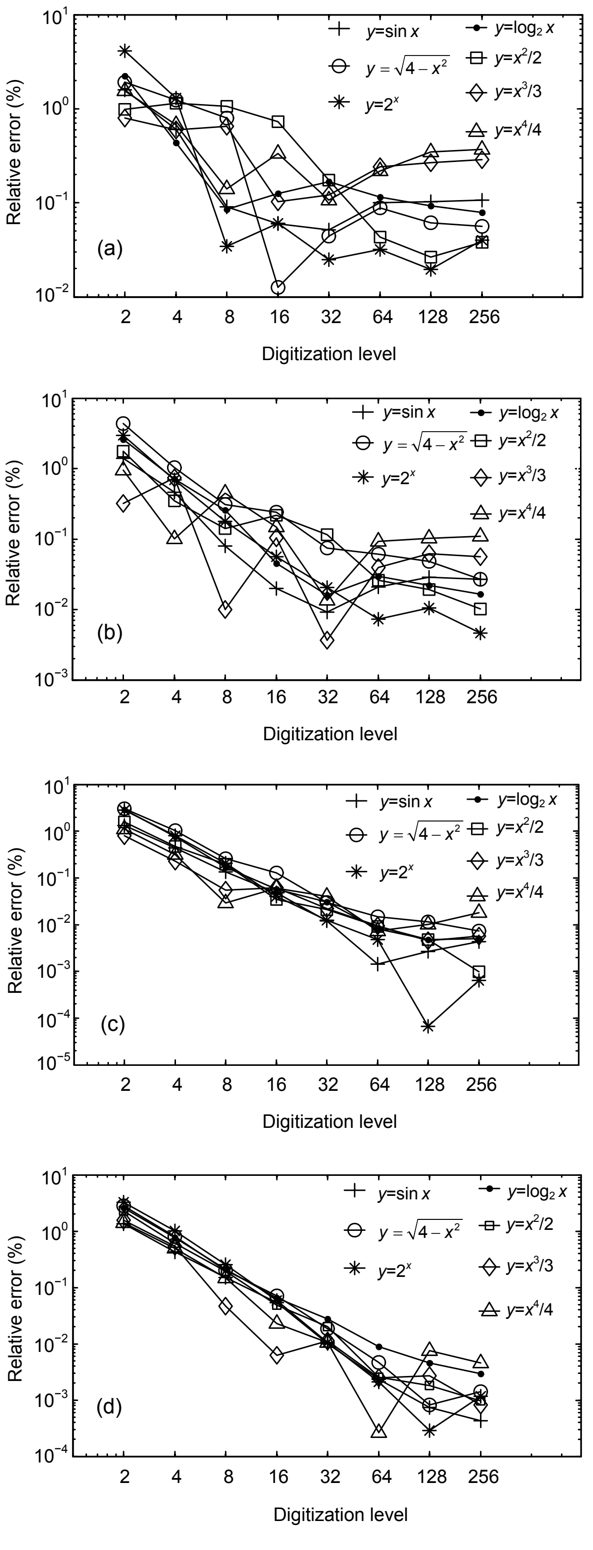

Fig. 6 Relative errors for average blurred elementary function images

(a) Grid resolution=10, $\varepsilon^{\max}$=4.126%; (b) Grid resolution=20, $\varepsilon^{\max}$=4.330%; (c) Grid resolution=50, $\varepsilon^{\max}$=3.014%; (d) Grid resolution=100, $\varepsilon^{\max}$=3.235%

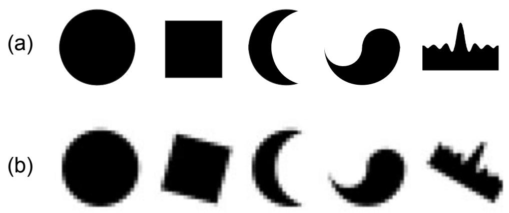

Fig. 7 Test shapes (Klette *et al.*, 1999)
(a) Vector graphics of the test shapes; (b) Pixel coverage digitization of the test shapes

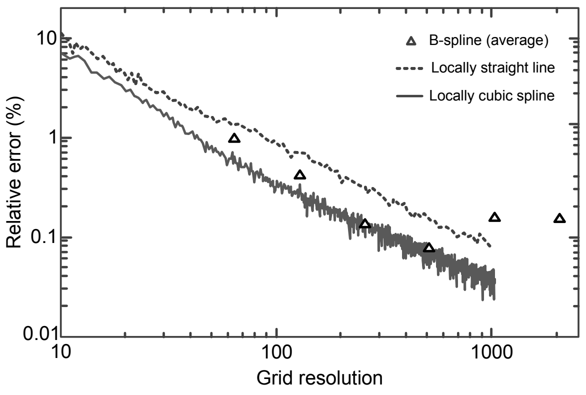

Fig. 8 Relative errors of the evaluation of B-spline (Suhadolnik *et al.*, 2010) (average relative error of grid resolutions 64, 128, 256, 512, 1024, and 2048), locally straight line (Sladoje and Lindblad, 2009), and locally cubic spline (our proposed model) on the test shapes given in Fig. 7

in which the relative errors at grid resolutions 1024 and 2048 are larger than those at grid resolutions 256 and 512, the local maximum relative error of our proposed model is always decreasing with increase of the grid resolution.

6 Conclusions

A novel local arc length estimator of the curve segment in the gray-scale image is presented. The method first estimates a cubic spline curve fit for the boundary points, and then calculates the approximate length of the spline segment between the knots by the first two terms of the Taylor series of the spline. The local arc length can be computed without explicitly knowing the spline polynomial coefficients, but only with the sums of y values. We present three local constraints on the proposed method. The pseudo code of the preprocessing and estimation algorithm is given, and the method can be implemented easily with linear complexity. To facilitate comparison with other published results, we test the proposed model using the same evaluation method as that used by Coeurjolly and Klette (2004) and Sladoje and Lindblad (2009). Test results show that our proposed estimator outperforms the other digital curve length estimators.

References

Cai, J., Walker, R., 2010. Height estimation from monocular image sequences using dynamic programming with explicit occlusions. *IET Comput. Vis.*, **4**(3):149-161. [doi:10.1049/iet-cvi.2009.0063]

Coeurjolly, D., Klette, R., 2004. A comparative evaluation of length estimators of digital curves. *IEEE Trans. Pattern Anal. Mach. Intell.*, **26**(2):252-258. [doi:10.1109/TPAMI.2004.1262194]

Deng, H., Himed, B., Wicks, M.C., 2006. Image feature based space-time processing for ground moving target detection. *IEEE Signal Process. Lett.*, **13**(4):216-219. [doi:10.1109/LSP.2005.863676]

Dyer, S.A., Dyer, J.S., 2001. Cubic-spline interpolation. 1. *IEEE Instrum. Meas. Mag.*, **4**(1):44-46. [doi:10.1109/5289.911175]

Foteinopoulos, P., 2009. Cubic spline interpolation to develop contours of large reservoirs and evaluate area and volume. *Adv. Eng. Softw.*, **40**(1):23-29. [doi:10.1016/j.advengsoft.2008.03.005]

Freeman, H., 1961. On the encoding of arbitrary geometric configurations. *IRE Trans. Electron. Comput.*, **10**(2):260-268. [doi:10.1109/TEC.1961.5219197]

Klette, R., Kovalevsky, V.V., Yip, B., 1999. Length estimation of digital curves. *SPIE*, **3811**:117-128. [doi:10.1117/12.364118]

Lachaud, J.O., Provençal, X., 2011. Two linear-time algorithms for computing the minimum length polygon of a digital contour. *Discr. Appl. Math.*, **159**(18):2229-2250. [doi:10.1016/j.dam.2011.08.002]

Sladoje, N., Lindblad, J., 2009. High-precision boundary length estimation by utilizing gray-level information. *IEEE Trans. Pattern Anal. Mach. Intell.*, **31**(2):357-363. [doi:10.1109/TPAMI.2008.184]

Suhadolnik, A., Petrišič, J., Kosel, F., 2010. Digital curve length calculation by using B-spline. *J. Math. Imag. Vis.*, **38**(2):132-138. [doi:10.1007/s10851-010-0208-4]

Tadrous, P.J., 2010. On the concept of objectivity in digital image analysis in pathology. *Pathology*, **42**(3):207-211. [doi:10.3109/00313021003641758]

Zhang, M., Mou, X., Zhang, L., 2011. Non-shift edge based ratio (NSER): an image quality assessment metric based on early vision features. *IEEE Signal Process. Lett.*, **18**(5):315-318. [doi:10.1109/LSP.2011.2127473]