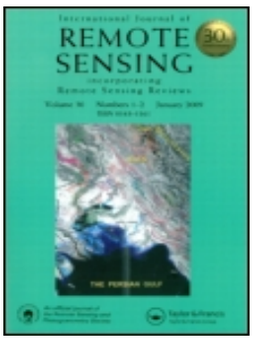


International Journal of Remote Sensing

ISSN: 0143-1161 (Print) 1366-5901 (Online) Journal homepage: http://www.tandfonline.com/loi/tres20

Linear feature extraction using adaptive least-squares template matching and a scalable slope edge model

Xiangyun Hu , Zuxun Zhang & Jonathan Li

To cite this article: Xiangyun Hu , Zuxun Zhang & Jonathan Li (2009) Linear feature extraction using adaptive least-squares template matching and a scalable slope edge model, International Journal of Remote Sensing, 30:13, 3393-3407, DOI: 10.1080/01431160802562198

To link to this article: http://dx.doi.org/10.1080/01431160802562198

Published online: 22 Jul 2009.

International Journal of Remote Sensing
Vol. 30, No. 13, 10 July 2009, 3393–3407

Taylor & Francis
Taylor & Francis Group

Linear feature extraction using adaptive least-squares template matching and a scalable slope edge model

XIANGYUN HU*†, ZUXUN ZHANG† and JONATHAN LI‡

†School of Remote Sensing and Information Engineering, Wuhan University, 129 Luoyu Road, Wuhan 430079, PR China

‡Department of Geography, Faculty of Environmental Studies, University of Waterloo, 200 University Avenue West, Waterloo, Ontario, N2L 3G1, Canada

(*Received 12 February 2007; in final form 10 July 2008*)

This paper presents a linear feature extraction method. Least squares template matching (LSTM) is adopted as the computational tool to fit the linear features with a scalable slope edge (SSE) model, which is based on an explicit function to define the blurred edge profile. In the SSE model, the magnitude of the grey gradient and the edge scale can be described by three parameters; additionally, the edge position can be obtained strictly by the 'zero crossing' location of the profile model. In our method the edge templates are locally and adaptively generated by estimating the three parameters via fitting the image patches with the model, accordingly the linear feature can be positioned with high accuracy by using LSTM. We derived the computational models to rectify straight line and spline curve features and tested those algorithms using the synthetic and real remotely sensed images. The experiments using synthetic images show that the method can position the linear features with the mean geometric error of pixel location of less than one pixel in certain noise levels. Examples of semiautomatic extraction of buildings and linear objects from real imagery are also given and demonstrate the potential of the method.

1. Introduction

Mathematically linear features can be described by straight lines or arbitrary curves (e.g. streams, shorelines, road centrelines, etc), which are usually vector data in digital mapping and geographic information systems (GIS). Linear features extracted from remotely sensed imagery have been an important data source for updating GIS vector data, as well as for applications including automatic and accurate shoreline extraction from imagery for monitoring dynamic changes of water level. Object boundaries projecting onto imagery are mostly linear features with edges reflecting the discontinuity of spectral or radiometric value of the pixels. Pixel size of remotely sensed imagery varies from less than 1 m to more than 1 km. Accurate location of edges can boost accuracy overall, so that precise linear features can be obtained, especially from low and intermediate resolution images. To date sub-pixel edge detection has been extensively studied, and moment-based methods have been frequently used (Tabatabai and Mitchell 1984, Lyvers *et al.* 1989, Pei and Cheng 1999, Shan and Boon 2000, Cheng and Wu 2005, Qu *et al.* 2005). Grey value moment or spatial moment has been employed to solve unknown edge model

*Corresponding author. Email: xiangyun.hu@gmail.com

International Journal of Remote Sensing
ISSN 0143-1161 print/ISSN 1366-5901 online
http://www.tandf.co.uk/journals
DOI: 10.1080/01431160802562198

parameters to indicate accurate edge location. The edge model may be an ideal step edge (e.g. Lyvers *et al.* 1989) or a blurred edge model (Shan and Boon 2000), which is more realistic than the ideal step model. As pointed out by Ye *et al.* (2005), the result of this method may have large errors for images with noise due to its implementation based on numerical differentiation. Thus the solution becomes extremely difficult due to the fact that this is a complicated model with additional parameters. Another method for sub-pixel edge extraction is based on interpolation and the analysis of scale behaviour of explicit geometric models for line profiles. Algorithms can be used for extracting lines and edges at the sub-pixel level (Steger 2000). Least squared error-based methods attempt to estimate edge location at the sub-pixel level by fitting the grey values of the image to an assumed edge model. Nalwa and Binford (1986) employed the hyperbolic tangent function for fitting the slope edge profile. Kisworo *et al.* (1994) presented an iterative procedure for evolving the initial position to the precise position for extracting step, ramp and roof edges. In their method, the error function is defined as the difference between the local energies of the model and the real data, as a function of the model parameters. One advantage of the least squared error-based method is that it is reliable for processing images with considerable noise. A comparison study using the moment-based, interpolation-based and least squared error-based methods can be found in Ye *et al.* (2005). Their experiments used synthetic and real images to show that the least squared error-based method can result in higher accuracy than the other two methods, especially for noisy images.

The above-mentioned methods could be seen as 'low-level processing' by which the edge (or line) pixels are accurately positioned through the use of a local image window (e.g. an $n \times n$ pixel window). Edge pixels are located individually to their sub-pixel locations. They have not integrated the geometric model of linear features into the computational model. In the last decade, several optimization-based methods have been proposed for extracting linear features by finding optimal 'routes' depending on global geometric and radiometric constraints of the feature. Dynamic programming (Grün and Li 1995), snake (or active contour models) methods (Trinder and Li 1995, Gunn and Nixon 1997, Nixon and Aguado 2001), least squares template matching (LSTM) (Grün and Agouris 1994, Hu *et al.* 2004, Kim *et al.* 2004) and a combined LSB-snake method (Grün and Li 1997) are essentially optimization-based methods that minimize 'cost' or 'energy' or 'least squared error' in order to obtain a reliable result. The LSTM method fits an image with the ideal feature model (template) for least squared error. LSTM has great advantages for integrating 'internal' (e.g. geometric) constraints and 'external' (e.g. image features) constraints. This applies rigorous theory as well as robust quality control and result assessment methodologies (Li 1997). Most of these proposed methods based on LSTM use simple edge or object (centreline) models, for example the step edge model. With these simple models, locating edge pixels by finding local maxima of gradient magnitude can only position the edge points to pixel level accuracy. Based on the inaccurate edge point locations it generates simple edge templates, which do not fit the blurring edge with varying scales; consequently it is not feasible to obtain high geometric accuracy since the edge scale (blurring) and noise level can vary significantly on real remotely sensed imagery.

Hough transform has been used for linear feature extraction for decades. There is the classical Hough transform (Duda and Hart 1972) and the generalized Hough transform (Ballard 1981), which can not only detect an object described with an

analytic equation (e.g. straight line, circle, etc), but can also detect an arbitrary object described with its model. By its nature Hough transform is not suitable for detecting linear features to sub-pixel accuracy. For instance, to obtain a parameterized sub-pixel straight line, the voting parameter space will be huge. On the other hand, Hough transform does not take into account the edge blurring and its scale varying factors. For some object extraction applications (e.g. shoreline mapping), particular techniques were employed. For instance, soft classification and super-resolution mapping techniques were used to accurately map the shoreline (Giles *et al.* 2005, Muslim *et al.* 2006). If an object can be delineated by its edge boundary, the rigorous edge model is still a feasible alternative measurement in order to locate the object to sub-pixel accuracy.

In this paper, we present a linear feature extraction method based on integrating the LSTM adjustment method with a scalable slope edge model for processing scale-varying (blurred) edges and images with considerable noise. Also combining with the geometric model of linear features as a whole, we derived computational models for rectifying straight line and spline curve features to their accurate position. The adjustment model is derived from continuous two-dimensional space. The essential rational of the method is to pull an initial inaccurate linear feature to its best fit by minimizing least square errors between the local scale-varying edge templates and corresponding image windows. The details of the algorithms are given in the following sections. We tested these algorithms using synthetic images and satellite images. The test results are evaluated and discussed.

2. Algorithm description

2.1 *Straight line rectification algorithm with LSTM*

The rectification of straight lines can be treated as a least squared error based method. As shown in figure 1, the solid line lies in the initial position and needs to be rectified to its exact position indicated by the dashed line. The length of the line is L. ΔY_0 and ΔY_1 are the shift along the normal direction at the end point of the line. Note that ΔY_0 and ΔY_1 are not by nature in pixel units: they are continuous values computed by the model to adjust the inaccurate line to its precise location. To a pixel P close to the initial line and located in x, which is the distance to the start point of

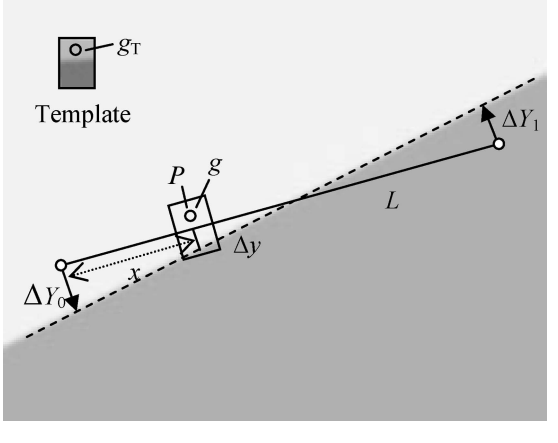


Figure 1. Rectification of a straight line.

the line, let its grey value be g and its geometric error in the normal direction be Δy. Assuming the edge template is already known and the grey value on the corresponding pixel is g_T, we have $g_T = g(y + \Delta y)$ and the error function:

$$v_g = \frac{\partial g}{\partial y}\Delta y - (g_T - g) = g_y \Delta y - l_g, \tag{1}$$

where g_y is the grey gradient in the normal direction. From:

$$\Delta y = (x/L)\Delta Y_0 + ((L-x)/L)\Delta Y_1 \tag{2}$$

and equation (1), we have

$$(x/L)g_y \Delta Y_0 + ((L-x)/L)g_y \Delta Y_1 = l_g. \tag{3}$$

This is the computational model for rectifying a line with adjustment processing. We can iteratively shift the line end point with (ΔY_0, ΔY_1) until the terminating condition is satisfied.

2.2 *Spline curve rectification algorithm with LSTM*

A cardinal spline is used for describing smooth curve features. The cardinal spline is a cubic Hermite spline whose tangents are defined by points and a tension parameter. The cardinal spline takes the positions of the current point and, along with the previous and next points, averages the positions using the tension value. This smoothes the line and makes a path that is gently curved through the points rather than zigzagging through them. As shown in figure 2, a piece of cardinal spline defined by the four continuous control points P_{i-1}, P_i, P_{i+1} and P_{i+2} by

$$\begin{aligned} P(u) &= P_{i-1}\left(-su^3 + 2su^2 - su\right) + P_i\left[(2-s)u^3 + (s-3)u^2 + 1\right] + P_{i+1}\left[(s-2)u^3\right. \\ &\left. + (3-2s)u^2 + su\right] + P_{i+2}\left(su^3 - su^2\right) = P_{i-1}C_0(u) + P_i C_1(u) + P_{i+1}C_2(u) + P_{i+2}C_3(u) \end{aligned}, \tag{4}$$

where u is the parameter, with $0 \leqslant u \leqslant 1$, $s=(1-t)/2$, and t is the tension value. To the pixel p where the error in the normal direction is Δy_N, we have

$$g_T = g(y_N + \Delta y_N) \tag{5}$$

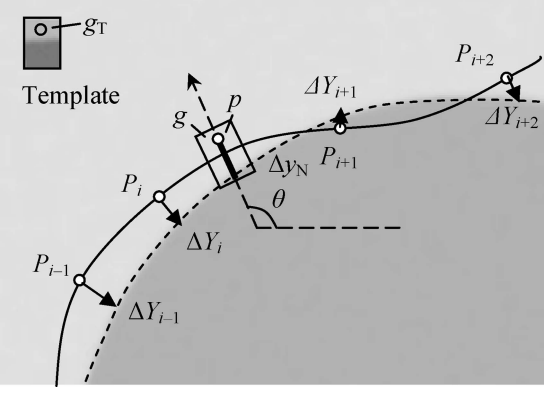


Figure 2. Rectification of a cardinal spline.

$$v_g = \frac{\partial g}{\partial y_N}\Delta y_N - (g_T - g) = g_{y_N}\Delta y_N - l_g, \tag{6}$$

where angle θ indicates the direction of the normal vector, $\Delta y_N = \Delta y/\cos\theta$, when $|\cos\theta| > |\sin\theta|$, or $\Delta y_N = -\Delta x/\sin\theta$, , when $|\cos\theta| \leqslant |\sin\theta|$, g_{y_N} is the grey value gradient in the normal direction. From equation (4), we know that the error of p in the x and y direction, Δx and Δy, can be described as:

$$\Delta y = C_0(u)\Delta Y_{i-1} + C_1(u)\Delta Y_i + C_2(u)\Delta Y_{i+1} + C_3(u)\Delta Y_{i+2} \tag{7}$$

$$\Delta x = C_0(u)\Delta X_{i-1} + C_1(u)\Delta X_i + C_2(u)\Delta X_{i+1} + C_3(u)\Delta X_{i+2}. \tag{8}$$

Based on the above equations, we get an adjustment model for rectifying the cardinal spline:

$$\sum_{n=i-1}^{i+2} C_{X(n)} \cdot \Delta X_n + \sum_{n=i-1}^{i+2} C_{Y(n)}\Delta Y_n = l_g, \tag{9}$$

where $C_{X(n)}$ and $C_{Y(n)}$ are the coefficients of the unknown adjusting variables ΔX_n and ΔY_n which indicate the geometric errors of the control points. The coefficients are determined by grey value of the pixels on the curve g, grey value of the edge template g_T, the grey value gradient g_{y_N} and the spline coefficients. For each pixel on the curve, if g_T is known, a least square adjustment equation (9) can be constructed; we call it one observation. Because the pixel number on a spline is far more than the number of its control points, the redundant observations are then employed for iteratively adjusting the initial curve (interpolated by the control points) to the precise location, which is described by equation (4) in two dimensional continuous space rather than in pixel grid space.

Thus far we have derived the computational models for rectifying the straight line and cardinal spline curve features. This could be useful for semi-automatic feature extraction, by which the initial position of the feature is provided in advance, or vector-image registration and vector data updating, for example the updating of shorelines from satellite images. In the next section, we focus on the using of a scalable slope edge (SSE) model, which is employed to obtain the grey value g_T of the edge template.

2.3 *A scalable slope edge model*

Figure 3 depicts the explicit model of the scalable slope edge profile, which is one-dimensional in the edge direction. The model is given as:

$$g(s) = h + \frac{k}{1 + \exp(-\alpha s)}, \tag{10}$$

where h and k are the intensity of the background and the contrast, respectively, and α controls the 'scale' of the edge. A larger α results in a steeper edge, while smaller α leads to a less steep edge. As shown in figure 3, the strict geometric position of the edge locates at the 'zero crossing' position, which is defined by the second derivative of the profile model $g''(s)$ or the maximum of the first derivative $g'(s)$. As discussed by Elder and Zucker (1998), the standard approach to edge detection is based on a

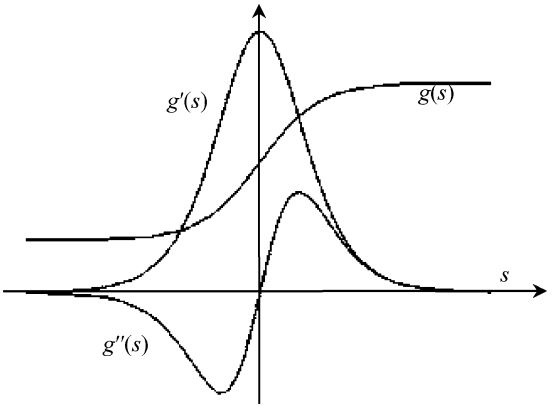


Figure 3. Scalable slope edge model depicting slope edge profile $g(s)$, the first derivative $g'(s)$ of $g(s)$ and the second derivative $g''(s)$ of $g(s)$.

model of edges as large-step changes in intensity. This approach fails to reliably detect and localize edges in natural images, where blur scale and contrast can vary over a broad range. There are a wide range of edge scales on remotely sensed images. This explicit model can be seen as an approximation to the Gaussian edge model employed in other applications (e.g. Elder and Zucker 1998, Shan and Boon 2000).

2.4 *Adaptive generation of edge template*

In order to integrate the edge model into the derived models of linear feature rectification, we have to automatically generate the edge template for each observation pixel. In other words, each edge template is generated locally and adaptively by estimating the three parameters indicating the intensity, contrast and scale of the edge on the pixel.

- Step 1. Generate the default edge templates $g_T(s)$ based on fixed h and k and varying α. For example, setting $h=50$, $k=50$ and $\alpha=2.0$, 1.0 and 0.5, we get three edge-profile templates corresponding to three different edge scales.
- Step 2. In the normal direction of the current point of the linear feature, slide the default template and do cross correlation with the corresponding local image window whose size is the same as the template. Find the maximal correlation coefficient ρ_m which means the best match, and record the corresponding α, let $\alpha_0=\alpha$. If $\rho_m<\rho_T$ (e.g. $\rho_T=0.80$), we discard the observation because of the low possibility for the existence of an edge. Note that we need to reverse the template $g_T(s)$ to $g_T(-s)$ and do cross correlation again to find ρ_m which is correspondent to the edge in the opposite edge direction. From the image window g_m correspondent to ρ_m, we can estimate the initial intensity h_0 by averaging the low grey value pixels and the initial contrast k_0 by subtracting h_0 from the average high grey value.
- Step 3. As shown in figure 4, we use the pixels in the local image window g_m for estimating the accurate edge model in the local position. According to the edge model given in equation (10), the error between the pixel g_m and the corresponding value in the local template g_T can be expressed as:

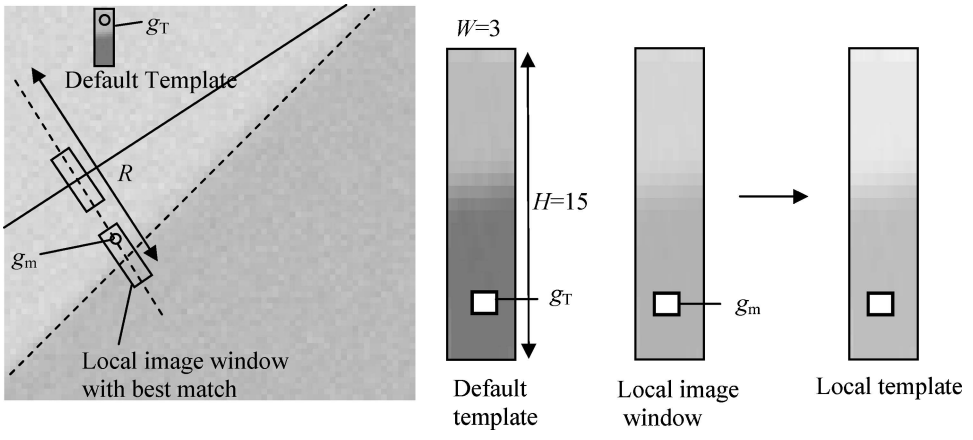


Figure 4. Adaptive generation of edge template. The default template is first used to match the local image window, then based on the best match the three parameters of the scaleable slope edge model are computed in order to obtain the local template, which is further used for estimating the errors of the two end points of the line.

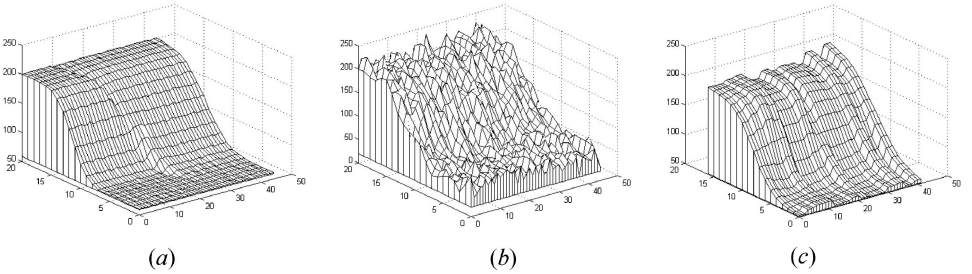
$$\Delta g = g_T - g_m = \Delta h + \frac{\Delta k}{1+\exp(-\alpha_0 s)} + \frac{k_0 \alpha_0 \exp(-\alpha_0 s)}{(1+\exp(-\alpha_0 s))^2}\Delta\alpha \tag{11}$$

This is the adjustment model for rectifying the initial parameters (h_0, k_0, α_0) to the least squared error-based method. The rectified model forms the template by the updated h, k and α. The adaptively generated template thus fits to the local intensity, contrast and the scale of the edge. Figure 5 shows an example of the adaptive generation of an edge template.

3. Implementation and experiments

3.1 *Implementation*

For the purpose of verifying the proposed method, the algorithms are coded using Microsoft Visual C++. As part of the process, several parameters have to be



(*a*) (*b*) (*c*)

Figure 5. An example of edge template generation. The axis of the horizontal plane is the image row and column, and the vertical axis is the grey value of the image pixel. (*a*) Gaussian blurred slope edge; (*b*) 10% NR Gaussian noise added to (*a*); (*c*) adaptively generated edge template from (*b*).

chosen. The iterative adjustment procedure described in §§2.1 and 2.2 is terminated when the maximal position error of the end points of the straight line or the control points of the cardinal spline are less than 0.02 pixels. To adaptively generate the edge templates, we have to do sliding correlation as described in §2.4. The search range of the sliding correlation limits the 'pull-in' range of the line rectification. In our implementation we set the search range $R=17$ pixels in the two opposite directions. We generate the default templates by setting $h=50$, $k=30$ and $\alpha=3.0$, 1.0, 0.6 and 0.4; so the four different scale templates serve as the initial templates. Our edge model is essentially a profile model along the edge direction but the image is a 2D array. Here we choose a W by H size to generate templates. $W=3$ is the width of the template and H is defined by the edge scale. In the implementation, we use a fixed scale of seven pixels to generate a template, so $H=2\times7+1=15$, and the edge position is centred in the template. The 3×15 size template is better than the one-dimensional template because it leads to more reliable results in cross correlation and estimation of the accurate edge model due to the abundant observations. For cardinal spline extraction, we use the tension value $t=0.5$. In these parameters, the search range constrains how far the line can be rectified. h, k and α initialize the gradient magnitude and scale of the edge template. Setting multiple α is to accelerate the process. If the scale of the real edge has a large difference with the initial template, it can take a great number of iterations to adjust it to the accurate one. By picking up one best fit of α, as in step 2 in §2.4, the time-consuming iterative adjustment process is accelerated. The selection of the size of the template is to make the template contain most of the edge information in its scale. Here we assume that the slope size of the edge is within seven pixels, for many real images it is appropriate based upon our observation. The actual initial values of h, k and α are not critical since they will be adjusted by the consequent LSTM.

3.2 *Experiments on synthetic images*

A synthetic image with dimensions of 256×256 pixels was created for straight line extraction. A diagonal edge is created by setting pixels on the opposite sides as low and high grey values with a range of 0–255, while setting the grey values on the diagonal line as 128. In this testing case, the accurate straight line is $y=x$ when the origin is put on the bottom left corner of the image, and the image row and column are the x and y axes respectively (figure 6). We change the contrast of the edge and blur the edge using Gaussian filtering based on various deviations ranging from 0.8

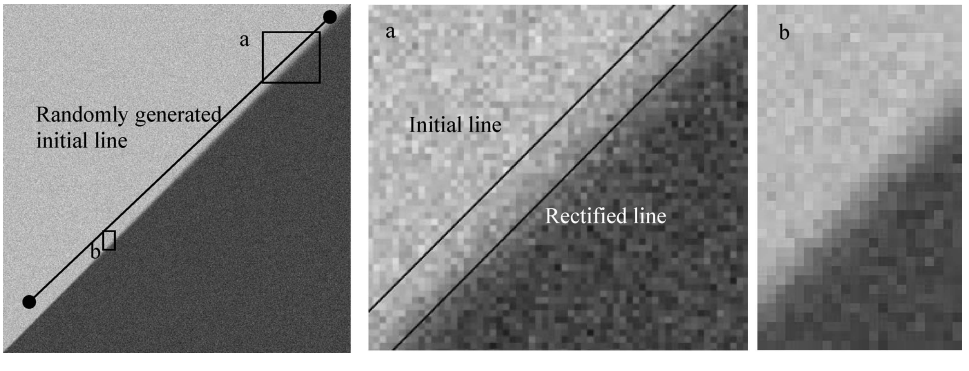


Figure 6. Test image for sub-pixel straight line extraction.

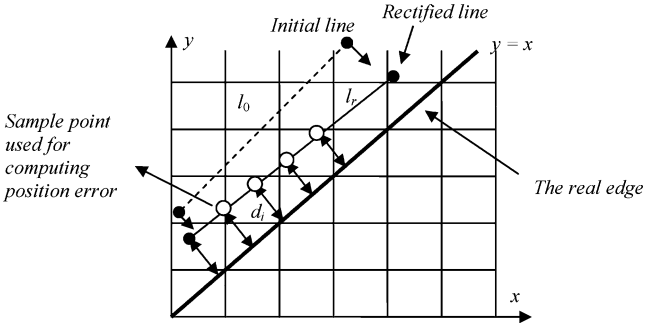


Figure 7. Estimation of per pixel accuracy for straight line rectification using the synthetic images. The x and y axes are the image row and column, respectively.

to 4.0 (along the edge and from lower-left to upper-right). Additive Gaussian noise is also included in four sets of images, with a mean of zero and a range of standard deviations related to noise ratios defined by Ye *et al.* (2005). The four different noise level images were created by different noise ratios: 5%, 10%, 15% and 20%; and 50 initial lines were randomly generated around the diagonal edge and in the search range. For each rectification, we compute the mean error $\bar{d}$ (per pixel) between the rectified line and the real line $y=x$. For each pixel on the line, the error can be computed by the distance from the point to the real line. Figure 6 displays the image with noise ratio=20% and the rectified result. The method of accuracy estimation is demonstrated in figure 7. $\bar{d}$ is the mean value of the distance from a pixel on the rectified line to the actual edge $y=x$ and l_0 is the initially placed inaccurate line. Letting the length of the rectified line segment l_r be L, we have

$$\bar{d} = \frac{\sum d_i}{L}. \tag{12}$$

d_i is the distance from the point on the rectified line to the real edge. The points employed to compute the distance are sampled on the rectified line in the interval of the pixel size, which is 1.0 in the two-dimensional image space. Figure 8 shows the mean error $\bar{d}$ on testing the four images. Depending on different noise levels, using the algorithm can rectify the straight line to the accurate position with a mean error of pixel position of less than 0.5 pixels.

We also tested curve extraction using equation (9). Similarly the two test images with 10% and 20% noise ratios were generated based on the original image, which was created by putting a solid circle with a radius of 100 pixels in the image centre and blurring the edge using Gaussian filtering, having different deviations ranging from 1.0 to 3.0. The initial curves were randomly generated, as shown in figure 9. The error d can be obtained by subtracting the radius from the distance from the point to the centre of the circle. For the 10% noise ratio image, the mean error $\bar{d}$=0.46 pixel, and the maximal d is 1.26. For the 20% noise ratio image, the mean error $\bar{d}$=0.54 pixel, and the maximal d is 1.44 pixel. Taking an example, from figure 10 we can see that most errors are less than 1.0, although the maximum may exceed 1.0. Results based on testing of noisy images indicate our method is effective. The relatively lower accuracy compared to the straight line rectification may result from the inaccurate representation of the circle using the cubic spline to fit the round shape.

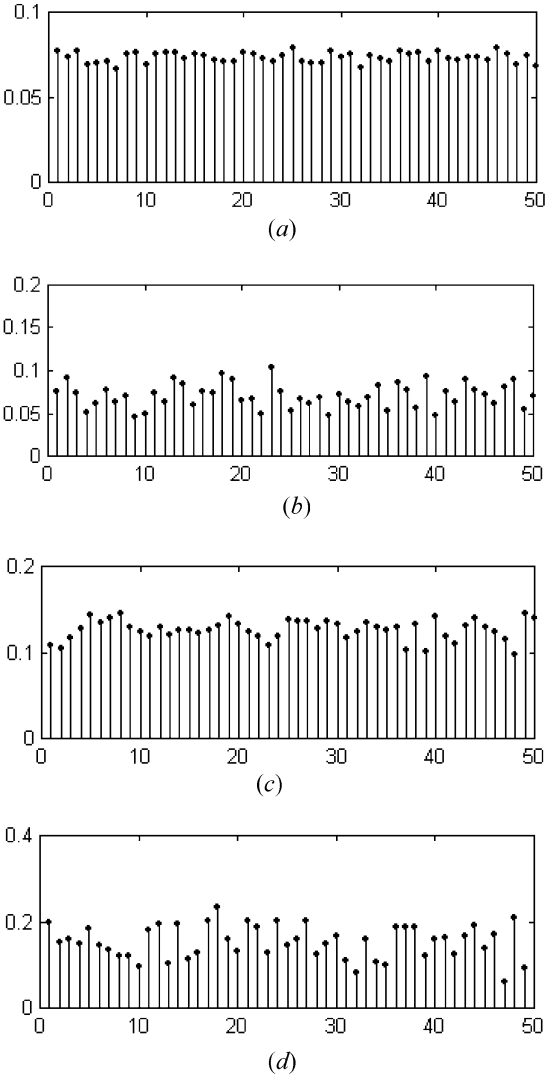


Figure 8. Mean errors of edge pixels on the straight line by 50 rectifications. The horizontal axis is the number of times that random initial lines were generated, while the vertical axis is the position error in pixels. (*a*) noise ratio=5%, mean of $\bar{d}$=0.07; (*b*) noise ratio=10%, mean of $\bar{d}$=0.07; (*c*) noise ratio=15%, mean of $\bar{d}$=0.12; (d) noise ratio=20%, mean of $\bar{d}$=0.15.

3.3 *Experiments on aerial and satellite images*

Figure 11 illustrates two examples of applying the straight line rectification algorithm to semi-automatically extracting building roofs from aerial images. The approximate building corner positions were provided (marked as crosses or polygon) by the human operator in advance. Then the developed computer algorithms rectify the lines which lead to the final precise delineation of the building roof outlines. Figure 11(*a*) shows the magnified local image window overlaid by the rectified lines. Visually we can find it located lines at the centre of the edges in varying blur scales. Figure 11(*b*) illustrates that the iterative LSTM process eventually adjusts the roughly placed initial lines to their precise locations.

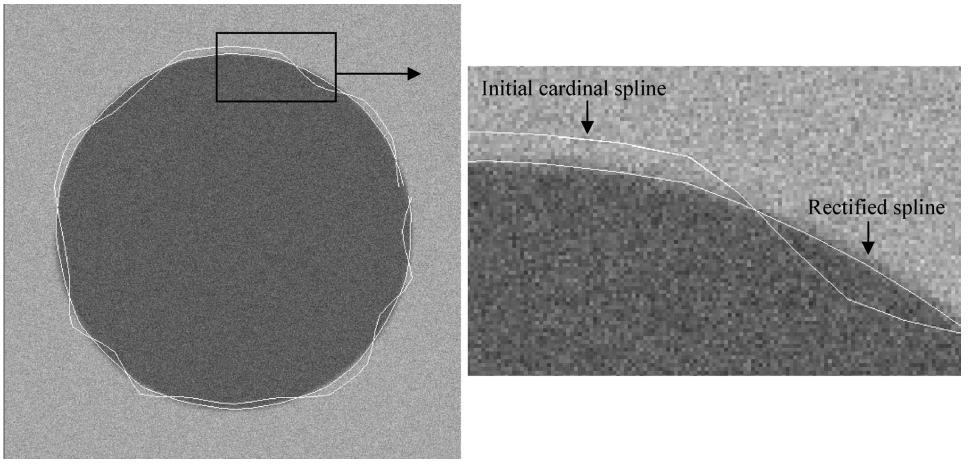


Figure 9. Test sub-pixel spline curve extraction on image with 10% of noise ratio.

Figure 12 shows the results of shoreline extraction from satellite images of different ground resolutions, with figure 12(*a*) the 23 m resolution IRS 1C image, and figure 12(*b*) the 4 m resolution Ikonos image. The initial shorelines are provided by manually placing the control points of Cardinal splines which are spatially close to the actual shorelines within the 'pull-in' range in template matching step. In the tests, we set the search range of the template matching as 17 pixels, which is about 391 m and 68 m in (*a*) and (*b*), respectively. The rectified curvilinear features demonstrate that the proposed algorithms can successfully extract the shorelines more precisely.

Theoretically, only if the transform function of the imaging system is known is rigorous localization of the physical edges from imagery achievable. Focal blur due to finite depth-of-field and penumbral blur at the edge of a shadow are major causes of blurred edges on the images taken by the optical camera. Since it has been argued

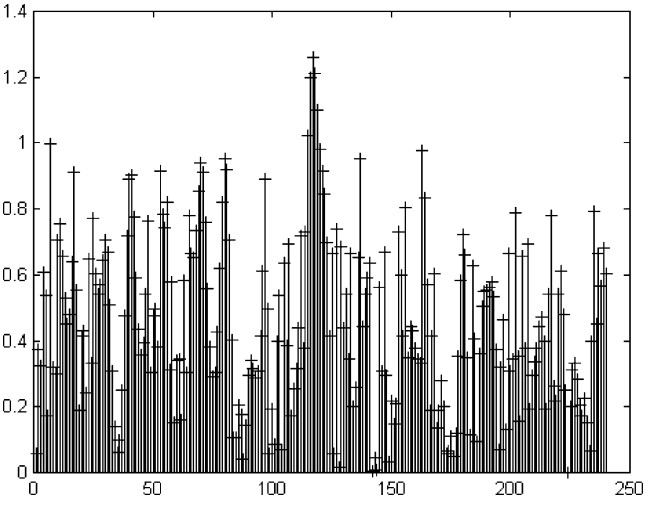
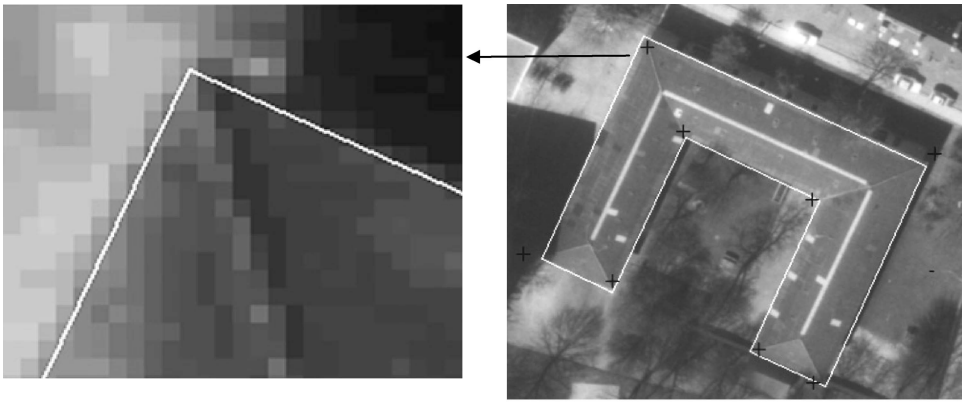
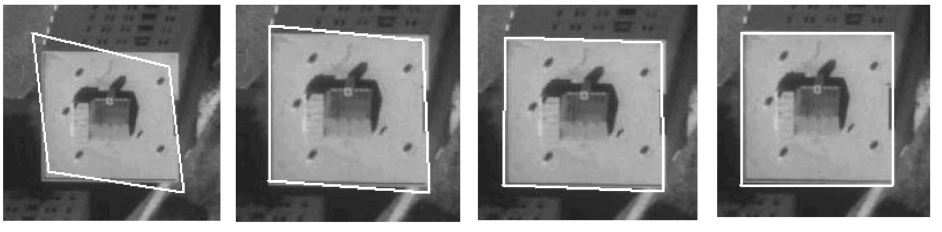


Figure 10. Errors on testing circle extraction based on the image with 10% of noise ratio. The horizontal axis is the point number, which is evenly resampled along the circle, while the vertical axis indicates the position error of the point.



(a)



Initial lines of roof top

First iteration, maximum corner point shift is 10.24 pixels

Third iteration, maximum corner point shift is 1.23 pixels

Thirteenth iteration, maximum corner point shift is 0.076 pixels

(b)

Figure 11. The results of building extraction from aerial image using straight line rectification: semi-automatic building extractions for (*a*) example 1; (*b*) example 2.

that the Gaussian blurring model better accounts for the various aberrations in real optical imaging systems, as Elder and Zucker (1998) used for edge detection in pixel level accuracy, we use the approximate Gaussian blurring model for setting up the edge template for LSTM in order to adaptively obtain exact edge template in varying scales. It is true that when using our method the same high accuracy may not be possible if it processes aerial or satellite images with very high noise levels and huge uncertainty in the edge profile model, etc. For a great number of usual images on which edges with sigmoid blurred profile indicate the object boundary, our method can be used to locate the features to sub-pixel accuracy in considerable noise levels and locally varying edge scales, as demonstrated in the experiments using synthetic images and real images.

On the other hand, so far we have only done visual inspections of the linear feature rectification by overlaying the rectified line vector and the magnified remotely sensed image. Quantitative evaluation of the rectification result requires accurate ground truth data (for instance, the shoreline curve at the photographing moment), which we do not yet have.

4. Conclusion and discussion

This paper presented a linear feature extraction method based on integration of the least squares template matching and a scalable slope edge model. The employment of

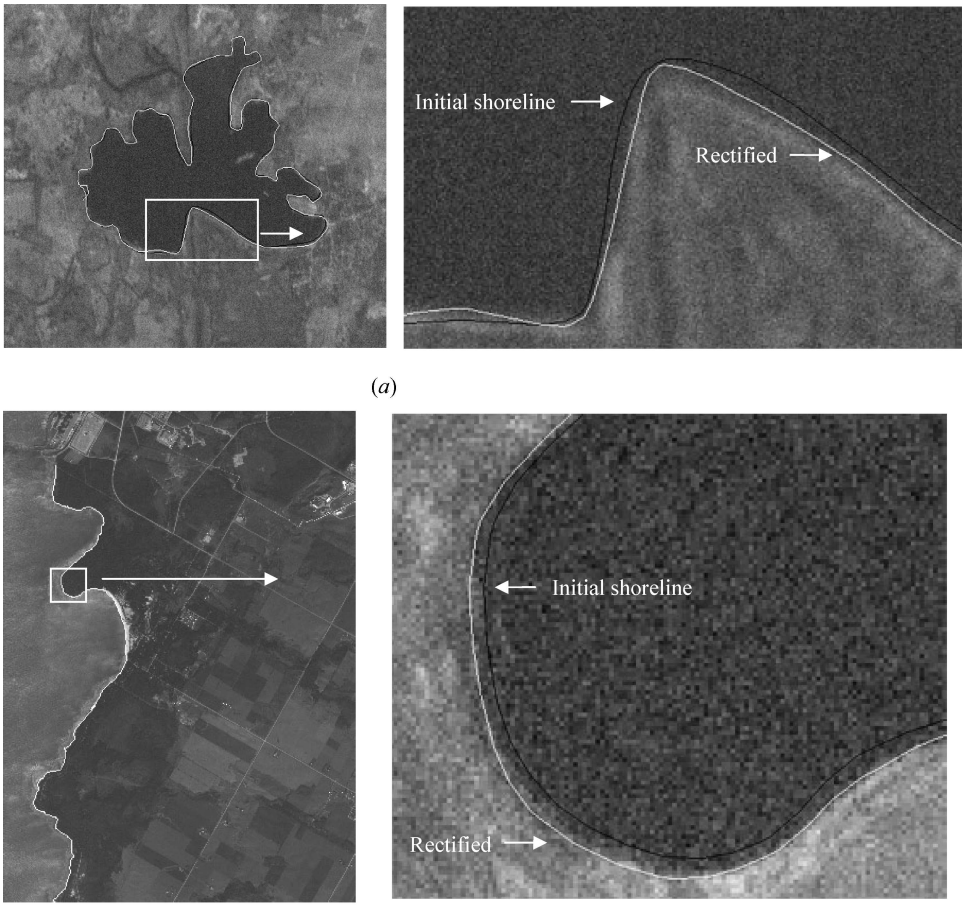


Figure 12. Examples of shoreline extraction from (*a*) 23 m IRS 1C and (*b*) 4 m Ikonos images.

the scalable edge model and adaptive edge template generation is advantageous for extracting lines when the edge varies in scale, intensity and contrast and when images contain noise. The scalable edge model is constructed by centring the edge position at the rigorous 'zero-crossing' position which is defined by the second derivative of the grey value profile. The geometric model of linear features can be constructed by initial or 'seed' lines placed by the human operator or from existing vector data. Then the algorithms can delineate curvilinear features precisely. The experimental results on the aerial and satellite images indicate that the proposed method can delineate both the building roof outlines and the shorelines robustly and efficiently.

Although experiments show the usefulness and potential of our algorithm in considerable image noise levels, given the varying image conditions and the restrictions imposed in the method, theoretically the algorithm does not solve the problem of locating linear features at sub-pixel accuracy. There are still limitations and issues that need to be addressed, especially for future improvement.

- The proposed adjustment models require initial position of the linear feature, which is described by straight line or spline curve. Only if the actual linear feature is spatially close to the initial line can the rectification result be reliable. For real world applications, using the proposed method, semiautomatic extraction with user-provided initial lines or feature updating with existing vector data are feasible. Fully automatic sub-pixel accuracy extraction based on our method depends on automatic detection of the initial features, which have yet to be studied.
- So far we only do quantitative evaluation of the geometric accuracy of the extraction based on synthetic images while assessing the result from real imagery by visual inspection, which involves magnifying the image and overlaying extracted lines. We need to do more strict evaluation of absolute geometric accuracy of the method using real remotely sensed images and high accuracy ground truth data.
- Extensive research on practical utility of our method should be carried out. We only consider slope edges as object boundary, which cannot cover all the real feature model types. For example, shore lines on images do not necessarily always demonstrate slope edges, noise levels can vary largely all over the image and the contextual objects can be very complicated. Object cues obtained by image classification (from multispectral images, etc) and other methods should be combined. Other further improvements include imposing geometric constraints (e.g. smoothness and three-dimensional conditions for object extraction from stereo images) on the model and deriving new image template to extract more linear features (e.g. road centrelines), etc.
- In our method of rectifying cardinal splines the tension value t was set a constant value. The influence of varying t has not been modelled. To adaptively determine its value to match the linear feature with imagery more precisely will introduce more computational cost. An efficient method of optimized selection of t has yet to be studied.

Acknowledgment

The work was supported by the National Key Technology R&D Program of China under grant No. 2006BAB10B01.

References

BALLARD, D.H., 1981, Generalizing the Hough transform to detect arbitrary shapes. *Pattern Recognition*, **13**, pp. 111–122.

CHENG, S.W. and WU, T.L., 2005, Subpixel edge detection of color images by principal axis analysis and moment-preserving principle. *Pattern Recognition*, **38**, pp. 527–537.

DUDA, R.O. and HART, P.E., 1972, Use of the Hough transforms to detect lines and curves in pictures. *Communications of the ACM*, **15**, pp. 327–336.

ELDER, J.H. and ZUCKER, S.W., 1998, Local scale control for edge detection and blur estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **20**, pp. 699–716.

GILES, F., MUSLIM, A. and ATKINSON, P., 2005, Super-resolution mapping of the waterline from remotely sensed data. *International Journal of Remote Sensing*, **26**, pp. 5381–5392.

GRÜN, A. and AGOURIS, P., 1994, Linear feature extraction by least squares template matching constrained by internal shape forces. *International Archives of Photogrammetry & Remote Sensing*, **30**, pp. 316–323.

GRÜN, A. and LI, H., 1995, Road extraction from aerial and satellite images by dynamic programming. *ISPRS Journal of Photogrammetry and Remote Sensing*, **50**, pp. 11–20.

GRÜN, A. and LI, H., 1997, Semi-automatic linear feature extraction by dynamic programming and LSB-snakes. *Photogrammetric Engineering & Remote Sensing*, **63**, pp. 985–995.

GUNN, S.R. and NIXON, M.S., 1997, A robust Snake implementation: a dual active contour. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **19**, pp. 63–68.

HU, X.Y., ZHANG, Z.X. and TAO, C.V., 2004, A robust method for semi-automatic extraction of road centerlines using a piecewise parabolic model and least squares template matching. *Photogrammetric Engineering & Remote Sensing*, **70**, pp. 1393–1398.

KIM, T., PARK, S.R., KIM, M.G., JEONG, S. and KIM, K.O., 2004, Tracking road centerlines from high resolution remote sensing images by least squares correlation matching. *Photogrammetric Engineering & Remote Sensing*, **70**, pp. 1417–1422.

KISWORO, M., VENKATESH, S. and WEST, G., 1994, Modeling edges at subpixel accuracy using the local energy approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **16**, pp. 405–410.

LI, H., 1997, Semi-automatic road extraction from satellite and aerial images. PhD. Dissertation, ETH Zurich, Switzerland.

LYVERS, E.P., MITCHELL, O.R., AKEY, M.L. and REEVES, A.P., 1989, Subpixel measurements using a moment-based edge operator. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **11**, pp. 1293–1309.

MUSLIM, A., GILES, F. and ATKINSON, P., 2006, Localized soft classification for super resolution mapping of the shoreline. *International Journal of Remote Sensing*, **27**, pp. 2271–2285.

NALWA, V.S. and BINFORD, T.O., 1986, On detecting edges. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **8**, pp. 699–714.

NIXON, M.S. and AGUADO, A.S., 2001, Feature extraction and image processing. (Oxford, UK: Newnes, Jordan Hill).

PEI, S.C. and CHENG, C.M., 1999, Color image processing by using binary quaternion-moment-preserving thresholding technique. *IEEE Transactions on Image Processing*, **8**, pp. 614–628.

QU, Y.D., CUI, C.S., CHEN, S.B. and LI, J.Q., 2005, A fast subpixel edge detection method using Sobel–Zernike moments operator. *Image and Vision Computing*, **23**, pp. 11–17.

SHAN, Y. and BOON, G.W., 2000, Sub-pixel location of edges with non-uniform blurring: a finite closed-form approach. *Image and Vision Computing*, **18**, pp. 1015–1023.

STEGER, C., 2000, Subpixel-precise extraction of lines and edges. *International Archives of Photogrammetry and Remote Sensing*, **XXXIII**(B3), pp. 141–156.

TABATABAI, A.J. and MITCHELL, O.R., 1984, Edge location to subpixel values in digital imagery. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **6**, pp. 188–201.

TRINDER, J. and LI, H., 1995, Semi-automatic feature extraction by snakes. In *Automatic Extraction of Man-Made Objects from Aerial and Space Images*, A. Grün, O. Kübler and P. Agouris (Eds), pp. 95–104 (Basel, Switzerland: Birkhäuser Verlag).

YE, J., FU, G.K. and POUDEL, U.P., 2005, High-accuracy edge detection with blurred edge model. *Image and Vision Computing*, **23**, pp. 453–467.