Experiments with Edge Detection using One-dimensional Surface Fitting

Gabor Terei, Jorge Luis Nunes e Silva Brito

The Ohio State University, Department of Geodetic Science and Surveying
1958 Neil Avenue, Columbus, Ohio 43210, USA

ABSTRACT

The approach of Nalwa and Binford for edge detection, using least squares surface fitting and a tanh basis for step-edges is described in detail in their paper [1]. The algorithm has been implemented, and tested for synthetic and aerial images. In this paper we are dealing with aspects of the algorithm that emerged during the implementation, and some aspects not, or only briefly mentioned in [1]. The algorithm is described shortly. We considered the thresholds for the changes of the unknowns in the nonlinear adjustments, as well as the maximum number of iterations allowed. We used the computed theoretical standard deviations to propose realistic limits. Also, the sensitivity of the adjustment to the initial values of the unknowns is discussed. Finally we present some results using the algorithm.

INTRODUCTION

Edge detection is one of the first steps in any procedure that has the goal of image understanding, due to the generally agreed upon fact that different objects are separated in the real world by color and texture, which then appear in the image as different color and texture, or different grey values, separated by edges. Indeed, two neighboring objects cannot be differentiated (by humans or computers), if they have the same, or very similar grey values. Thus the first step is to extract, or at least enhance the edges in the image.

An edge in an image is a discontinuity in the intensity (usually grey values) of the image, where the intensities on both sides of the edge have relatively distinct properties. These discontinuities can be mapped into three general categories based upon their appearance. Probably the most common edges are step-edges, although line edges and roof edges also appear in the images. One can also differentiate between physical edges corresponding to abrupt changes in the surface normal and characteristics in object space, and optical edges which correspond to abrupt changes in grey value, color, or texture in the image. Unfortunately an optical edge does not always correspond to a physical edge, and a physical edge does not always appear as an optical edge. During image processing, such as edge detection, we can only detect optical edges, which would then need to be further analyzed to relate them to a physical edge, making the linking of the edges an important factor.

Most of the edge detection algorithms developed are based on one of two schemes. One is the detection of local extremes of the gradient of the image, while the other relies on the calculation of the Laplacian of the Gaussian, and subsequent detection of the zero-crossings. Both schemes are discussed in more detail in textbooks on digital image

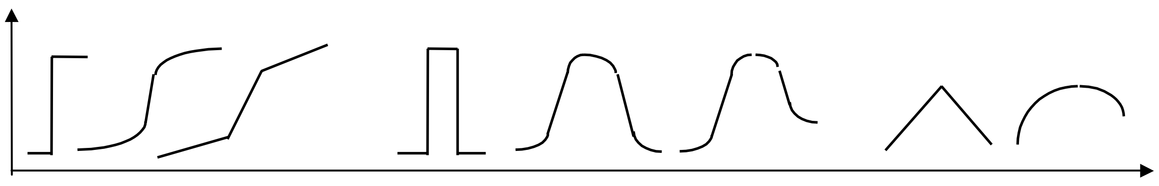


Figure 1. Step-edges, line-edges, and roof-edges. The first step-edge is an ideal edge, the second is ideal for this specific model.

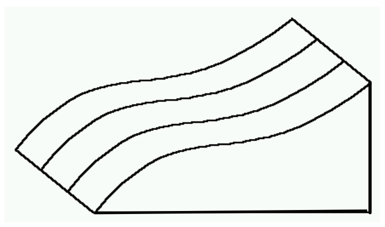


Figure 2. One-dimensional surface

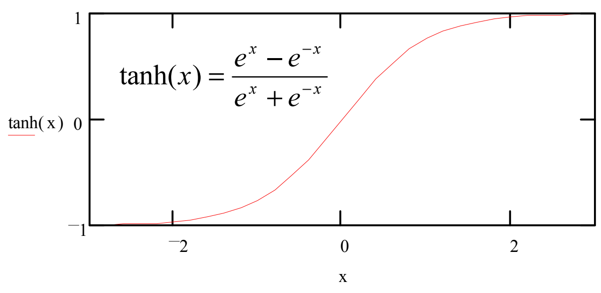


Figure 3. The tanh function

processing [2], and computer vision [3], with several examples and development ideas. The main problem associated with taking the derivatives is that they emphasize high frequency noise, resulting in false edges.

The approach developed by Nalwa and Binford [1] is conceptually different from the above. Their idea is to fit a series of one-dimensional surfaces to a grey value window and accept the surface description that best approximates the local grey value function in the least squares sense. A one-dimensional surface is a surface that is constrained to be constant along a specific direction (Figure 2). Instead of searching for pixels that are defined as edge pixels, this approach looks for edgels with an associated position and direction. An extended edge in the image is approximated by short linear segments called edgels, which are detected in the window. According to Nalwa and Binford, the algorithm can take into account the blurring effect of the imaging system, produces effective noise reduction without blurring the edges as severely as circularly symmetric operators (e.g., Gaussian) do, reduces the significance of thresholding, and requires only four parameters, resulting in a smaller window size, thus a better resolution.

To test the algorithm for photogrammetric use, we implemented it and ran it with synthetic and aerial imagery. In this paper we are dealing with aspects of the algorithm that emerged during the implementation, providing test results, and elaborating on some aspects not, or only briefly mentioned in [1].

DESCRIPTION OF THE ALGORITHM

The algorithm described by Nalwa and Binford deals only with step-edges, which are the most dominant type, and this is the version we have implemented. The tanh function (Figure 3) has been found as an adequate basis for step-edges, but combinations of the tanh function produce adequate basis for line and roof edges as well. The adequacy of the tanh function is described in detail in [1].

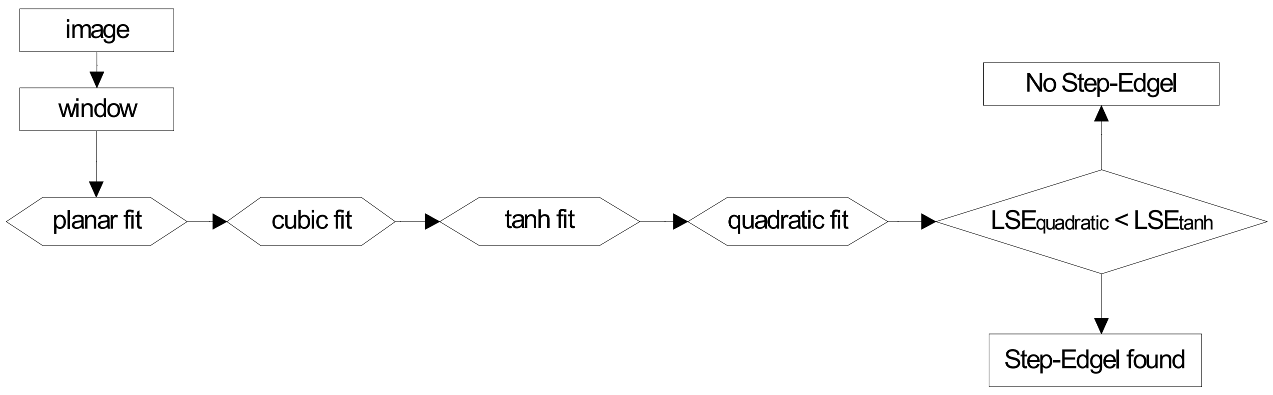


Figure 4. The algorithm as developed by Nalwa and Binford

As with most edge detection algorithms, the scheme is to select a window of the image, and perform analysis on that window to detect the presence of an edgel. A flowchart of the algorithm is shown in Figure 4. First a plane is fit to the window in the least squares sense. The aspect of this plane gives the first approximation for the direction of variation in the window, and further one-dimensional surfaces are constrained using this direction. The edgel (if detected) will be perpendicular to this direction. To refine this estimate, a one-dimensional cubic surface is fit to the window. A one-dimensional cubic surface is a surface constrained to be constant in one direction, and described by a cubic function in the orthogonal direction. The edge direction resulting from the cubic fit is used as the directional constraint for further surfaces. A 1-D tanh fit is then calculated in the least squares sense, giving the position of the edgel. A useful byproduct of the tanh surface is the edge contrast, or the step size, which is useful in linking and interpretation, and can be compared to a threshold for eliminating small steps. In the governing equation, '*f*' is a scaling factor introduced by Nalwa and Binford to take care of the blurring effect of the imaging system described by a Gaussian of σ_{blur}, as will be discussed later. Finally to discriminate against smooth shading, and eliminate non-step-edgels, a quadratic fit is performed, and the least squares error of the tanh, and the quadratic fit are compared. If the quadratic fit (which is an unconstrained basis with the same number of parameters as the tanh basis) describes the surface better, thus resulting in a lower least squares error, the edgel is considered to be a non-step-edgel, and is dropped.

EXPERIMENTATION

Implementation considerations

The observation equations for the planar and quadratic fits are linear, however, the cubic and tanh least squares adjustments are nonlinear in one of their parameters. The planar fit is performed to provide a reasonable initial value for the angle in the cubic fit equations. However, Nalwa and Binford do not elaborate on what initial values should, or could be used for the other parameters. By choosing good initial parameters, the number of necessary iterations can be greatly reduced, thus speeding up this computationally expensive algorithm, not to mention the sensitivity for convergence of some parameters to the initial values.

type of adjustment	observation equations	unknowns	comments
planar	$v = (a_0 + a_1 \cdot x + a_2 \cdot y) - I[x,y]$	a_0, a_1, a_2	linear
cubic	$v = (a_0 + a_1 \cdot z + a_2 \cdot z^2 + a_3 \cdot z^3) - I[x,y]$	a_0, a_1, a_2, a_3, θ	non-linear in θ ($z = z(\theta)$)
tanh	$v = (s \cdot \tanh(f \cdot [z + p]) + k) - I[x,y]$	s, p, k	non-linear in p
quadratic	$v = (a_0 + a_1 \cdot z + a_2 \cdot z^2) - I[x,y]$	a_0, a_1, a_2	linear

$$z = x \cdot \cos\theta + y \cdot \sin\theta \qquad f = \frac{0.85}{\sigma_{blur}}$$

Table 1. Recapitulation of the least squares adjustments involved in the algorithm

There are no problems with the planar fit, as it is a linear problem, and no initial values for the unknowns are needed. The only consideration is to detect homogeneous windows, where the angle is not defined. In this case, the whole process for the window can be stopped, as there evidently are no edges in a homogeneous area. For the cubic fit the equations are nonlinear in the angle, however, due to the initial estimate from the planar fit, convergence requires only a few steps. The initial values for the other parameters of the cubic function are once again arbitrary, and do not significantly affect the number of iterations required. The most sensitive case is the tanh fit, where the equations to be solved are nonlinear in ‘p’ (reflecting the position of the edgel), and the pull-in range is fairly small. Furthermore, our experiments showed that the pull-in range for ‘p’ not only depends on the initial value of ‘p’, but also on the good approximation of the parameters ‘s’ and ‘k’. If ‘s’ and ‘k’ do not reflect well the brightness and contrast of the step-edgel prior to the adjustment, ‘p’ needs to be known to around a quarter of a pixel, in some extreme cases even better. This is not acceptable, even though we know that the edgel has to pass through the center pixel of the window. We must be able to calculate ‘p’ in a range of ± 0.5 to $\pm\sqrt{2}/2$ pixels from the center of the window, depending on the angle of the edgel. The problem could be solved by offsetting the initial value of ‘p’ by a certain small amount and restarting the adjustment computations anew, if the least squares solution does not converge. However, this would require even more computations in this already computationally expensive algorithm. Instead, initial values for ‘s’ and ‘k’ should be derived from the results of the cubic fit. In the general case, the one-dimensional cubic surface fit to the window has a local maximum and local minimum in the window, which can be calculated easily, and used as the approximation of the intensities on both sides of the edgel. However, due to round-off errors, no real minimum and maximum may exist. In this case, we simply used the largest and the smallest values in the window to approximate ‘s’ and ‘k’. Basically, both approaches give sufficient approximation that the pull-in range of ‘p’ is expanded to about 0.75 pixel, which is sufficient. The initial value of ‘p’ is then set up in such a way, that the edgel would pass right through the center of the window, using the direction found in the cubic fit. The last least squares adjustment, the quadratic fit, is a linear problem, not requiring initial values and iterations.

Another consideration is when to stop the iterations: what should be the threshold for the change in the parameters? To answer this question, we performed tests using several windows extracted from real and synthetic images, calculated the accuracy figures for the parameters from the adjustment, compared the “true” values (where they could be determined) to the obtained values, and investigated if the accuracy estimates were correct. For the cubic surface, the standard deviations of the parameters a_0, a_1, a_2, and a_3

vary greatly, however, they are usually greater than one. Also, taking into consideration that we are looking for local extremes in a window, where x and y are small values (range from 0 to 4 in a 5x5 window), small changes in these parameters will not have a large effect on these extremes, which we will only be using as initial values for the tanh fit anyway. Thus a good rule of thumb is to stop the iterations once the change in the parameters becomes smaller than one. For the threshold on the change in the angle, our experiments show that the standard deviation indicates an accuracy lower than one degree, however the "real" values usually differ from the calculated values by more than a degree. This is mainly due to the quantization error, introduced by calculating a continuous function from discrete values, and we can accept Nalwa and Binford's suggestion to round the angle to the nearest 5^o. However, not rounding the angle has very little effect on the results of the tanh, and quadratic fits. This then means that we can stop the iteration once the change in the angle is less than 2.5^o. Using these thresholds, usually two iterations are sufficient.

As expected, in the case of the tanh fit, the accuracy figures depend largely on the step size, and the blurring of the step-edge. Figure 5 shows the edgels used for testing. We have used ideal step-edges, blurred step-edges, and extracted different quality edges from an aerial image for this purpose. Table 2 summarizes the results obtained during the least squares adjustment. Note that the positional accuracy turned out to be better then 0.1 pixels in all cases, except for a small step size edgel. In all cases, the comparison of the "true" values to the calculated values proves that the accuracy estimates are correct. The "true" values cannot be calculated for the edgels extracted from the aerial image, however, visual analysis shows that the results reflect the reality. Accuracy figures once again depended on the step size, as well as the clarity of the edgel. Taking these results into consideration, we suggest to use the accuracy figures from the least squares adjustment to determine the thresholds for stopping the iterations. We calculated the standard deviations of the parameters, and stopped the iteration if the changes in all of the parameters were below one half of their corresponding standard deviation. If the standard deviations for 'k' and 's' are below 0.5 (grey values), and below 0.1 (pixels) for 'p', we also stop the iterations, since realistically no better results can be achieved. Using these guidelines, usually only a few iterations were needed.

0	0	0	255	255
0	0	0	255	255
0	0	0	255	255
0	0	0	255	255
0	0	0	255	255

a

0	1	43	213	255
0	1	43	213	255
0	1	43	213	255
0	1	43	213	255
0	1	43	213	255

b

0	0	0	0	0
0	0	0	0	255
0	0	0	255	255
0	0	255	255	255
0	0	0	0	0

c

241	244	244	242	246
137	198	244	246	245
97	125	153	207	244
66	77	102	146	155
63	71	81	111	137

d

174	162	118	72	75
172	165	112	69	71
170	158	104	68	71
168	160	94	66	72
172	158	87	68	70

e

117	144	211	211	216
101	117	192	192	217
116	110	176	176	218
118	101	125	125	214
113	99	107	149	202

f

180	181	172	181	182
180	184	181	193	191
187	187	178	180	176
176	184	195	190	201
205	203	198	189	196

g

Figure 5. Edgels used for testing (a., ideal step-edge, b., ideal step-edge for the model, c., corner (no edgel), d., e., strong edges, f., well visible edge, g., just visible edge)

edgel code	# of iter.	k_{calc}	s_{calc}	p_{calc}	σ_k	σ_s	σ_p	k_{true}	s_{true}	p_{true}
id255	10	136.2	146.5	2.6	8.8	7.5	0.1	127.5	127.5	2.5
id128	10	59.7	74.5	2.4	4.4	3.8	0.1	64	64	2.5
id5	10	2.3	2.9	2.4	0.2	0.1	0.1	2.5	2.5	2.5
bl255	3	129.7	132.0	2.5	1.1	1.0	0.1	127.5	127.5	2.5
bl128	3	64.8	66.4	2.5	0.5	0.4	0.0	64	64	2.5
bl5	3	2.5	2.5	2.5	0.0	0.0	0.0	2.5	2.5	2.5
strong	2	165.6	81.7	0.7	4.2	3.7	0.1	-	-	-
good	4	120.2	51.8	1.9	0.9	0.8	0.0	-	-	-
med	2	162.1	56.23	1.1	2.2	2.0	0.1	-	-	-
poor	3	190.8	8.8	3.1	2.3	2.2	0.3	-	-	-

Table 2. Results and accuracy figures from the tanh fit. Edgel codes:
id*nnn* represents the ideal edge shown in figure 5a, *nnn* being the higher values,
bl*nnn* represents the blurred edge shown in figure 5b, *nnn* being the higher values,
strong, good, med, and poor are the windows extracted from the aerial image, shown in figure 5d-g.

In some cases, where the tanh model does not reflect the values in the window at all (e.g., Figure 5.c), the least squares adjustment will provide very slow convergence, or the parameters will oscillate around some value. Such a case slows down the edge detection process, and raises the question: how many iterations should be allowed? Oscillation, or slow convergence proves that the model does not fit the image well, and the window does not contain a step-edgel. Considering the results obtained from our tests, we have decided to limit the maximum number of iterations to 10, which was the maximum number of iterations (below 100) that were needed to reach convergence in any of our tests using the previously discussed thresholds.

Test images and experiments

To test the overall behavior of the algorithm, we have used a synthetic and an aerial image (Figure 6). The synthetic image was 256 by 256 pixels, while the aerial image was 512 by 512 pixels, both being 8 bit images. Table 3 summarizes the experiments performed. To test the smoothing effect of the algorithm, a version of the synthetic image with added noise was also used. As mentioned in the introduction and the presentation of the algorithm, to take the blurring effect of the imaging system into account, the scaling factor '*f*' is introduced in the tanh equation, which depends on the standard deviation of the effective blurring Gaussian during image acquisition. This blurring effect can be attributed to several factors, mainly atmospheric conditions, lens effects and motion blur, which cannot be determined exactly. Nalwa and Binford describe why this scaling factor is used in [1]. To test the effectiveness of the model, we also created a version of the noiseless synthetic image, blurred with a Gaussian of known σ, and ran the algorithm using different values for σ_{blur}. We also used different window sizes for the edge detection, from which we expected a better smoothing effect, especially on the noisy and the aerial image. However, there is a limit, defined by the image content, on the maximum window size, as the assumption of edgels being linear segments breaks down with large window sizes. The smallest window size, as explained in [1] is 5 by 5 pixels, which was used in most cases.

Image used	window sizes	σ_{blur}
synthetic without noise	5, 9	0.1, 0.5, 0.6, 0.7, 1.0, 1.2, 1.3, 1.4
synthetic with noise	5, 9	0.1, 0.6
synthetic blurred with $\sigma_{blur} = 0.6$	5	0.1, 0.5, 0.6, 0.7, 1.0, 1.2, 1.3, 1.4
aerial	5, 9, 15	0.4, 0.6, 0.8, 1.0

Table 3. Experiments performed on the test images

Results

The results of our test are shown in Figure 7. We can observe that the synthetic image was processed perfectly, and the results with the noisy synthetic image, as well as the results with the aerial image are satisfactory. Figure 7b shows the output generated from the noisy synthetic image. Here, the value of the grey level output corresponds to the calculated step size. The images shown in Figure 7 have been inverted for better representation, thus darker points correspond to larger step sizes. We can see that most of the noise appears as small step sizes, and can be eliminated by choosing an appropriate threshold. The threshold can either be determined interactively, or using statistical values based on the whole output image. By using a larger window size (figure 7c, binary output), the smoothing effect of the surface fitting is larger, however the corners are cut off more and more as we increase the window size. This effect is even worse in the aerial

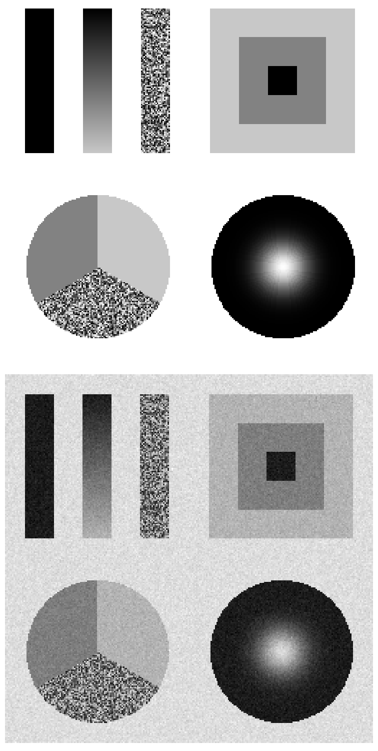
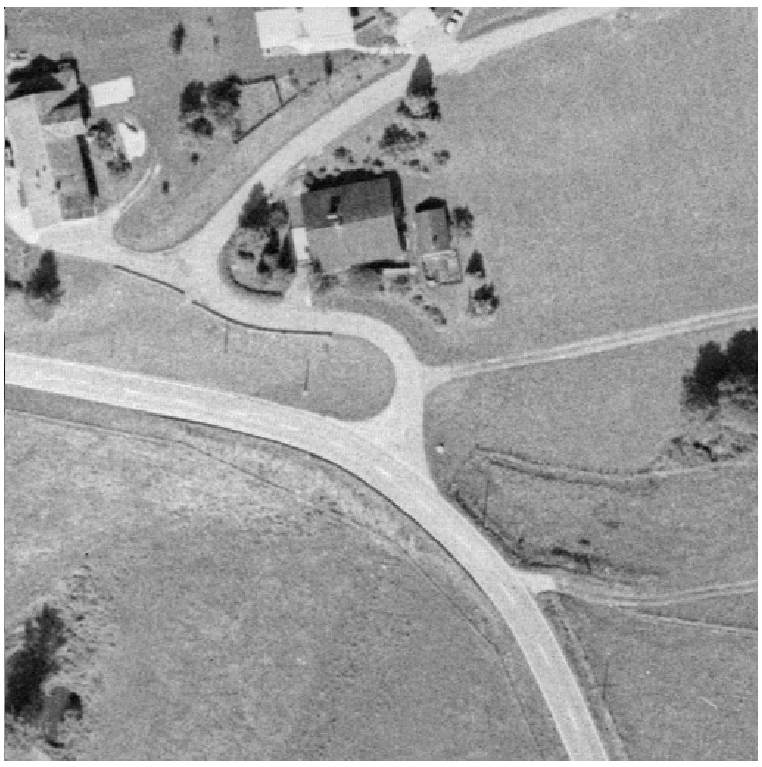
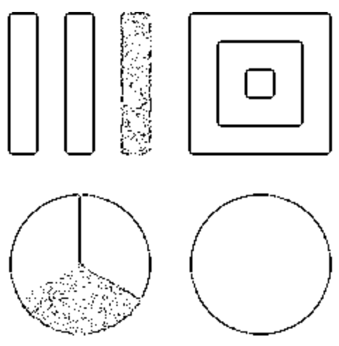
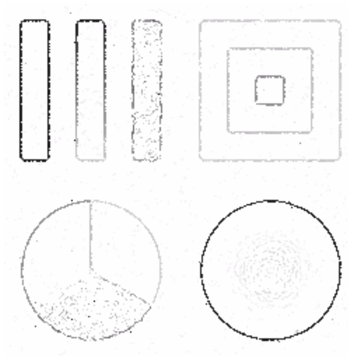



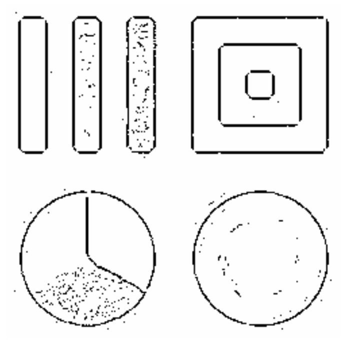
Figure 6. The test images. The noisy version of the synthetic image was produced by adding 10% random noise to the image.

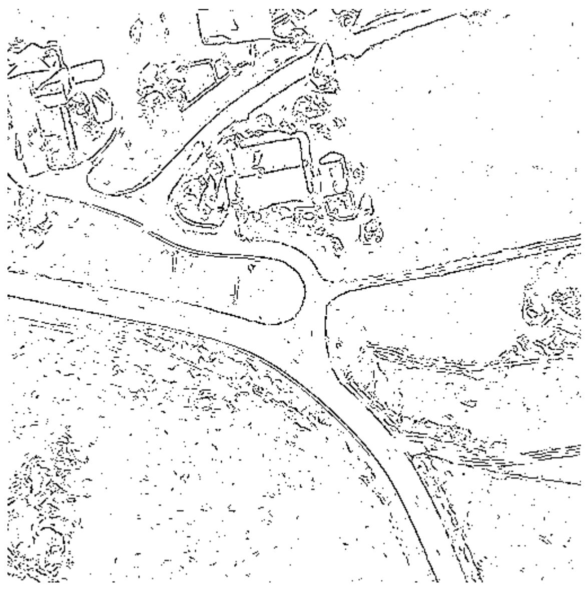
image, where sharp curves are not detected any more. The step-edges in the aerial image are detected very well, and the grey level output (figure 7d) shows that most of the texture and noise is smoothed out, or appears as very small step size edgels.

Other results not shown here prove that the effect of the blurring Gaussian is fairly low on the resulting output. In fact, changing the value of σ_{blur} in the image, or the algorithm by 0.1 results in changes of the decisive parameters that are less than the accuracy of these parameters. For real (aerial) images, any value for σ_{blur} between 0.4 and 1.0 gives good, basically indistinguishable results. For the synthetic image, using a σ_{blur} of 0 would be ideal, however that would result in division by 0. Using 0.1 gives perfect results, the only difference from the previous being in the accuracy of the step size calculation. Visible degradation occurred only with a σ_{blur} larger than 1.4, where the edges became shifted by a pixel. For the synthetic image, blurred with the Gaussian prior to edge detection, the step sizes of the edges are calculated correctly (within 2 grey values), proving that the tanh function is a good basis for ideally blurred step-edges.


a.,


b.,


c.,


d.,

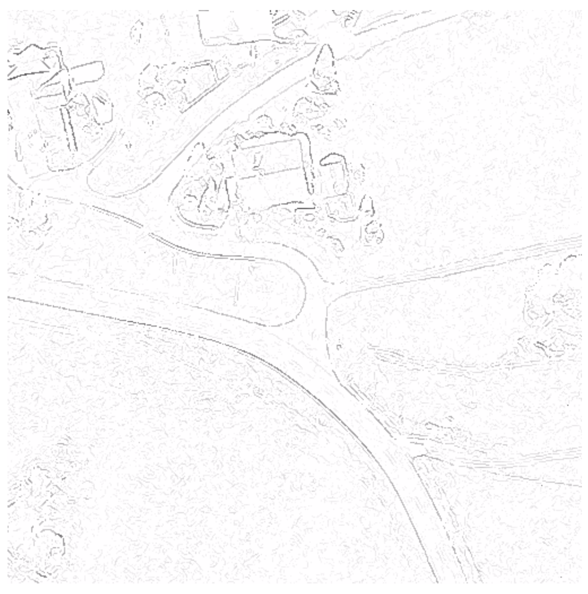

e.,

Figure 7. a., default output (synthetic image without noise), b., output as grey values (noisy synthetic image), c., window size 9 x 9 (noisy synthetic image), d., default output from the aerial image, e., grey level output of the aerial image

CONCLUSIONS

The overall performance of the algorithm is very satisfactory, and the only major drawback is its slow speed, due to the computational intensity of the task. With careful programming considerations, this speed can probably be increased, but it will still be about a magnitude slower than most other edge detection algorithms. The algorithm, however, carries many advantages, which makes it a considerable choice if speed is not of primary concern. The fact that it provides directional and contrast information is valuable for linking and interpretation. The accuracy figures obtained from the least squares adjustment can also be used for linking. For example, the standard deviation of the step size gives important information about the "quality" of the edge, and can be used for determining a threshold when linking an adjacent edgel. Sub-pixel accuracy is reached in the position, generally around 0.1 pixels. Another advantage is that there is no implicit thresholding in the algorithm itself, only the one associated with the step size, which is performed after the computations are done.

ACKNOWLEDGMENT

We would like to thank Dr. Christian Heipke for sacrificing his time to help us answer many of our questions, and for his support in the realization and publication of this paper.

REFERENCES

[1] Nalwa, V.S. and Binford, T.O., 1986. On Detecting Edges, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-8. No. 6, pp. 699 - 714.

[2] Gonzalez, R.C. and Woods, R.E., 1992. Digital Image Processing, Addison-Wesley Publishing Co. Inc.

[3] Haralick, R.M. and Shapiro, L.G., 1992. Computer and Robot Vision, Addison-Wesley Publishing Co. Inc.