A computational approach for edge linking

Ovidiu Ghita*, and Paul F. Whelan
Vision Systems Laboratory, School of Electronic Engineering, Dublin City University,
Dublin 9, Ireland

Efficient edge operators such as those based on partial derivatives fail to return continuous edge maps. To address this, a supplementary edge linking step is required to complete the initial edge information. In this paper we propose a fast and efficient algorithm for edge linking using the local information around edge terminators. In order to minimise incorrect linking decisions, the direction and the linking path for each edge terminator is established by minimising a cost function. The particular novelty of this approach lies in the labelling scheme which assigns the directionality of the edge terminators (endpoints) based only on local knowledge. As a consequence, it relaxes the demand of *a priori* knowledge and furthermore assures an accurate and efficient search for edge paths in the image.

Keywords: Edge detectors; Endpoints; Cost function; Edge linking.

1. Introduction

Successful edge detection is one of the most important steps in a wide range of image processing and analysis operations. While edges are associated with sharp changes in pixel intensity distribution, usually they are extracted by applying partial derivatives to the intensity image [4,9,15]. However, the edge map returned by the edge operators based on first and second derivatives fail to correctly recover the entire edge structure associated with a given image. In this regard, the recovered edge map either contains false edge points which are generated by image noise or exhibits gaps in edges due to a low variation

*Corresponding author. Tel.: +353-1-7005869; Fax: +353-1-7005508; E-mail: ghitao@eeng.dcu.ie

in the pixel intensity distribution. Thus, after edge extraction further processing has to be applied in order to eliminate the spurious edge responses and to link the gaps in edges.

Various techniques which address the problem of improving the quality of the initial edge detection have been investigated. Approaches that have been used include morphological methods [1,5,16,23], Hough transform [7], probabilistic relaxation techniques [8], multiresolution methods [2,14] and the use of additional information such as colour [18]. In general, morphological approaches offer a fast solution as they attempt to maximally exploit the local information. In this regard, Snyder et al. [20] proposed to close the gaps in the edges using a chamfer map-based algorithm. After the distance transform is computed around the edge structure, the surrounding pixels are re-labelled into edge pixels using a best neighbor strategy. This edge linking scheme proved to be efficient to close the gaps between closely spaced unconnected edges. Xie [25] proposed a Causal Neighborhood Window algorithm where the the horizontal edge segments were considered as seeds for edge linking. The main advantage of this edge linking algorithm is its low computational overhead but the performance is poor when dealing with highly textured scenes. Later, Hajjar and Chen [10] proposed a real time edge linking algorithm based on a VLSI architecture. In their formulation, the break points were detected and the gaps in edges are bridged according with the smallest distance between two compatible break points. Although simple, their algorithm significantly increases the level of connected pixels in the edge structure. As it is based on a hardware architecture the size of the scanning window is fixed and as a consequence their implementation does not guarantee to produce closed contours.

In contrast, multiresolution and multiscale methods attempt to enhance the edge structure by aggregating the information contained in a stack of images with different spatial resolutions. These methods also referred to as pyramidal techniques usually outperform morphological techniques, but this is obtained at a high computational cost. In this sense, Farag and Delp [6] proposed a multiresolution approach to edge linking using a sequential search algorithm. In their implementation they use a multiresolution image pyramid which allows the edge information contained in low resolution images to guide the sequential search at higher resolutions. Their algorithm proved to return high quality

connected edge contours when applied to arbitrary textured images. One problem with this approach is the large number of parameters that have to be tuned and furthermore they tend to be image dependent. Vincken et al. [24] proposed a multiscale linking model for image segmentation. In their implementation they used a hyperstack where the multiscale images are obtained by applying a linear diffusion model based on Gaussian filtering. Initially introduced by Perona and Malik [17], this approach has the advantage that it offers scale invariance which assures a precise location of edges across various scales, a fact that greatly simplifies the edge linking process. This algorithm was applied to the segmentation of brain images and the results indicate the efficiency of this strategy.

This paper presents a method of producing connected edge structures which is appropriate for use in conjunction with scene understanding algorithms. The algorithm firstly applies an efficient edge operator which initially blurs the image using an exponential function in order to remove image noise. This is followed by the edge linking scheme which closes the gaps in edges by analysing the local information around edge terminators. In order to minimise the incorrect linking decisions, the edge terminator is labelled by analysing the local edge structure and this information is used to determine the linking path in the unconnected edge structure. The devised edge linking algorithm produces quality connected edge maps at a low computational cost. A synopsis of the developed edge linking algorithm is illustrated in Figure 1.

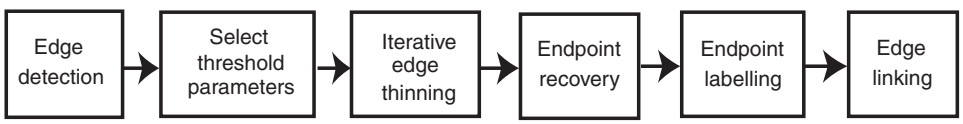

Figure 1. Outline of the proposed edge linking algorithm.

2. Edge detection

The image processing and analysis applications such as edge matching or object recognition are highly influenced by the quality of the edge detector employed. Robust edge detection is a difficult problem since the input image is affected by noise and the objects that define the scene exhibit a low contrast around their borders. To address these problems Canny [4] introduced the gradient of the Gaussian where the Gaussian operator is

applied in order to reduce spurious edge responses. To further improve the edge detected output, he implemented a simple algorithm based on thresholding with hysteresis in order to join the weak edge points with the edge structure defined by step-like edges. This algorithm evaluates the output of the edge detector using two threshold parameters which are often referred to as low and high thresholds. If the magnitude of an edge response is greater than the high threshold then it is assigned as an edge point. Then, the remaining edge responses are marked as edge points if their magnitude is situated between the low and high threshold values and follow the edge structure with the magnitude above the high threshold.

Although the gradient of Gaussian operator is a powerful edge detection technique it entails a high computational cost. Thus, the implementation outlined in this paper employs an optimal edge detector based on ISEF (Infinite Symmetric Exponential Filter) [19] where the computational overhead is one order lower than that required by the Canny edge detector. It is also important to note that the performance of the ISEF edge operator is closely matched to that offered by the Canny edge detector. Our experimental results are in line with those reported by Heath et al. [12] where the performances of a large number of edge operators are evaluated. Figure 2 depicts the performance of the Canny and ISEF edge operators when applied to a standard test image.

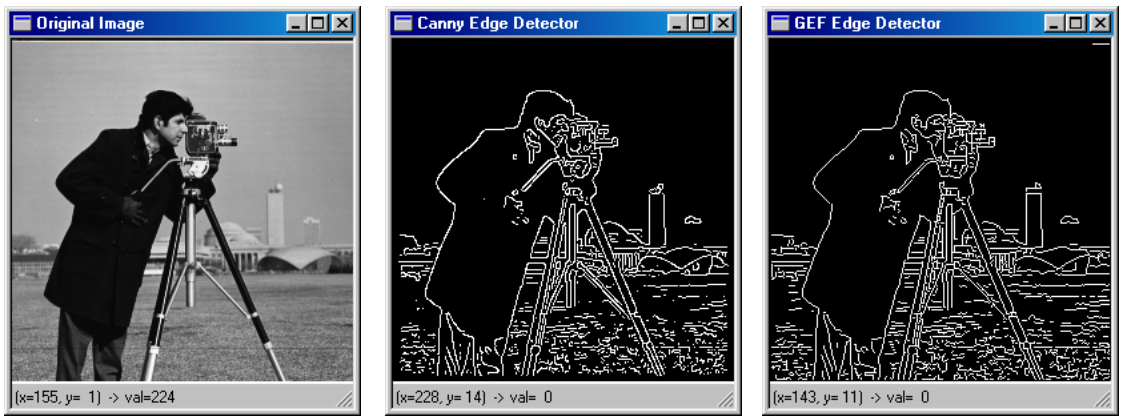

Figure 2. Performance of the Canny edge detector (σ=1.0, low threshold=30, high threshold=55) and the performance of the ISEF edge operator(a_0=0.45, low threshold=5, high threshold=7) when applied to the standard 'Cameraman' test image.

In line with the choice of the edge operator, detecting the optimal range of the internal parameters of the edge operator is an important issue. For our implementation we set the scale parameter for the ISEF edge operator to the default value (a_0=0.45) and the threshold parameters are selected by evaluating the level of small edge segments in the edge detected output. To do this, we employed a strategy where the high threshold is directly linked with the low threshold (i.e. the high threshold offsets the low threshold by an experimentally determined constant value). The thresholds are determined by evaluating the ratio between the number of pixels derived from small edge segments and the number of pixels derived from large segments which has to be smaller than a preset value. This is achieved by analysing if the level of small edge segments returned by the threshold pair with the minimum value for the low threshold respects the preset condition. If not, the value of the low threshold is increased and the algorithm verifies if the threshold selection condition is upheld for the new threshold pair. This procedure is iteratively applied until the level of small edge segments respects the threshold selection criterion.

3. Iterative edge thinning

Multiple edge responses represent another typical error associated with edge detecting operators. Thus, there are cases when the edge responses are several pixels wide. Since our goal consists of reconnecting the interrupted edges using only the local information, multiple edge replications may generate incorrect linking decisions. Therefore to use the local information efficiently, a thinning algorithm has to be applied to remove the unnecessary edge responses. To address this issue, an iterative morphological thinning algorithm based on the use of L-type structuring elements was implemented [21]. This algorithm is defined as follows:

$$I \oslash S = I - (I \otimes S) \tag{1}$$

where I is the image containing the edge information, S is the structuring element, $\oslash$ denotes the thinning operation and $\otimes$ defines the binary hit or miss transformation. The thinning process is convergent and stops when two successive images in the sequence are identical.

4. Endpoint recovery and labelling

Although the proposed scheme significantly improves the edge structure, there are situations where gaps in edges exist in the output image. To correct this problem we propose a method to close the gaps by analysing the singular edge points which are referred to as endpoints. Extracting the endpoints entails a simple morphological analysis [22] and consists of a set of 3×3 masks that are applied to the resultant image after the application of the edge detection and thinning process.

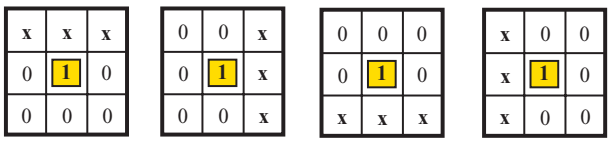

Figure 3. The masks used to detect the endpoints.

Figure 3 illustrates the masks used to detect the endpoints, where the pixel under investigation is highlighted and mask entries indicated by 'x' can take any value (0 or 1) but at least one of them has the value 1. This ensures that the single edge pixels are not marked as endpoints.

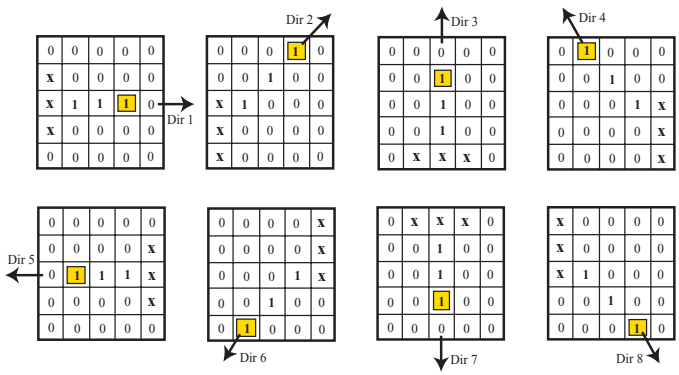

Figure 4. Situations where the edge direction (indicated with an arrow) is derived from straight edges.

To efficiently close the gaps in edges, requires the maximum exploitation of local knowledge. Thus, we need to determine the scanning direction for each endpoint by evaluating the linked edge pixels that generate it. As can be easily observed, the masks illustrated

in Figure 3 contain some information that gives a useful clue regarding the endpoint direction. Unfortunately, this gives only 4 scanning directions which is not sufficient to find always the correct result. To avoid such limitations we extend the search for edge links to 8 directions, a situation where supplementary information has to be evaluated. As Figure 4 illustrates, there are cases when the endpoint is generated by a straight edge, a situation where the scanning direction can be easily established.

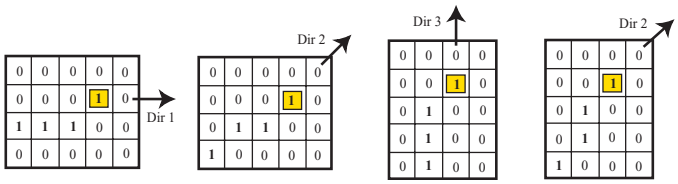

Figure 5. The edge direction derived from curved edges.

This may not be the case for curved edges, when the edge direction is not as well defined. A typical situation is illustrated in Figure 5 where the endpoint direction is evaluated by analysing the local information for a larger neighborhood. In Figure 5 only the first 3 directions are analysed, while the remaining directions can be obtained by rotating the masks.

5. Edge linking

The last step of the algorithm attempts to find possible edge paths using the information associated with the edge terminators. In this regard, the algorithm investigates the edge pixels situated at the side indicated by the endpoint direction. The endpoint neighborhood is a parameter of the algorithm and defines the size of the scanning window, which corresponds to the maximum linking distance. Thus, for each edge pixel situated in the endpoint neighborhood a linking factor is computed using a simple cost function. The cost function was designed to favor the edges that are close together and have opposite directions. For farther edges the linking coefficient returned by the cost function increases rapidly.

$$\Gamma(ep) = k_d dist(et, ep) + k_e + k_{dir} \tag{2}$$

where et, ep are the image co-ordinates of the edge terminator and the edge pixel, $dist(et, ep)$ is the Euclidean distance, k_e is a reward factor if the edge pixel is also an endpoint and k_{dir} is a reward factor if the edge pixel is an endpoint and its direction is opposite to that of the endpoint under investigation. The reward coefficients are determined experimentally and in our implementation we set k_d=0.25, k_e=0.5 if the edge pixel is an endpoint, 1 otherwise, k_{dir}=0.5, 1, 1.5, 2.0; k_{dir} takes a value of 0.5 when the directions of the endpoint and the linking endpoint are opposite, 1, 1.5 according to the deviation from the ideal case, 2.0 if the edge linking pixel is not an endpoint or the linking endpoint has a similar direction with the endpoint in question.

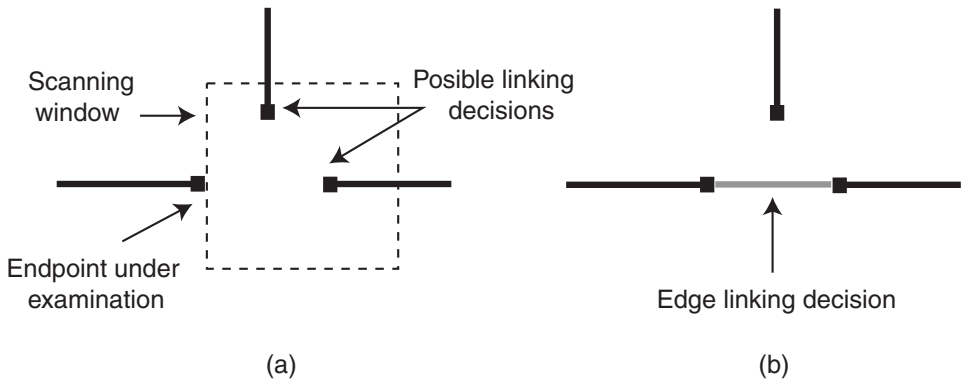

Figure 6. Edge linking process. (a) Typical unlinked edge structure. (b) Edge linking results after the cost function is evaluated.

After the cost function is computed for each edge pixel situated in the endpoint neighborhood, the minimal value determines the linking path and a line is drawn between the edge terminator and the edge pixel using the Bresenham algorithm [3]. Figure 6 depicts the edge linking process when dealing with a typical unlinked edge structure. The pseudo-code describing the Bresenham algorithm is outlined in Figure 7.

6. Experiments and results

The results of the complete edge linking process when the algorithm is applied to a noiseless standard test image is depicted in Figure 8. As Figure 8 illustrates, the algorithm was able to handle difficult situations such as edge bifurcation.

1. Plot the start point (x1,y1).
2. Compute diferences between the co-ordinates of the start point and the last point:
   dx = x2 - x1, dy = y2 - y1;
3. Initialize decision parameter:
   D_0 = 2dy - dx;
4. For each x_k along the line verify:
   if D_k < 0, then
      D_k+1 = D_k + 2dy, Plot_Pixel(x_k+1,y_k);
   else
      D_k+1 = D_k + 2dy - 2dx, Plot_Pixel(x_k+1,y_k+1);
5. If step 4 is executed dx times then the line drawing process is complete.

Figure 7. Pseudo-code describing the Bresenham line drawing algorithm.

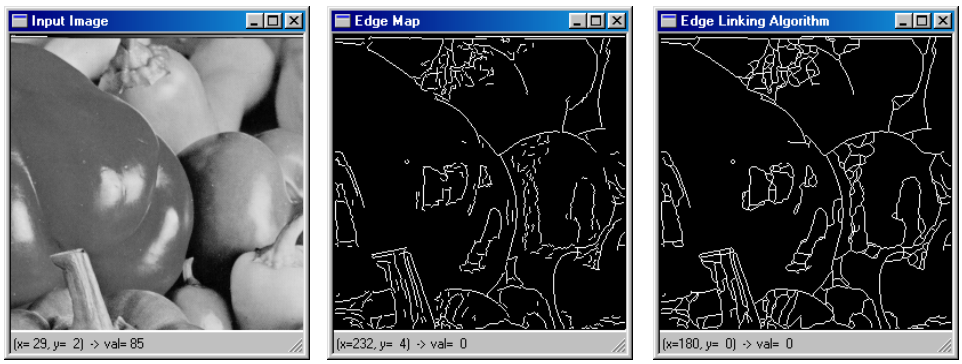

Figure 8. (a) The input image noiseless image. (b) The edge map returned by ISEF edge detector. (c) Edge linking results.

To verify the algorithm's robustness to noise, the image illustrated in Figure 8 was corrupted with additive Gaussian noise of standard deviation 30 grey-levels. In Figure 9 the result of the edge linking process is illustrated. It can be noticed that the algorithm was able to handle this complication and the result returned by the edge linking algorithm is almost identical with that depicted in Figure 8.

Next, to verify the validity of the proposed algorithm, we tested its performance on a range of images defined by various scenes and the results were compared with those returned by the Multiresolution Sequential Edge Linking algorithm (M-SEL) [6].

Figures 10 and 11 show the result of the proposed edge linking algorithm and the M-SEL algorithm when applied to a set of test images.

By analysing the results depicted above it can be noticed that the performance of

Figure 9. (a) The input image corrupted with Gaussian noise (standard deviation 30 grey-levels). (b) The edge map returned by ISEF edge detector. (c) Edge linking results.

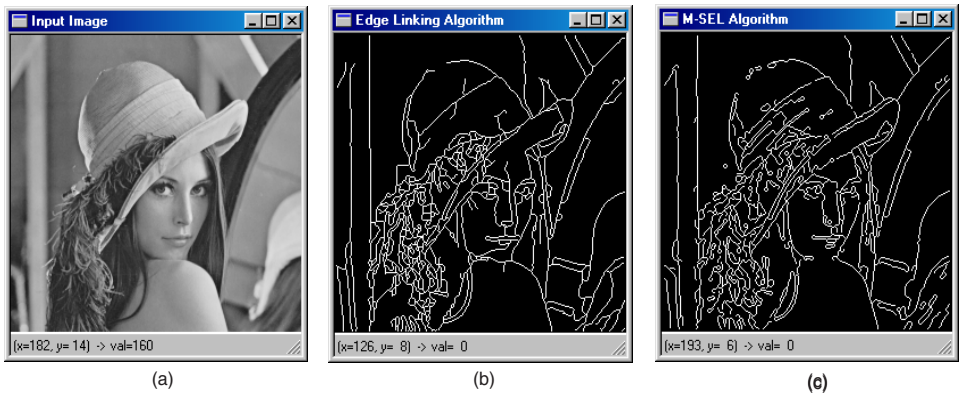

Figure 10. (a) The input image. (b) Edge linking results returned by the proposed algorithm (scanning widow 11×11). (c) Edge linking results returned by the M-SEL algorithm.

the proposed algorithm compares well with that offered by the more complex M-SEL algorithm. It can be observed that the edge linking strategy presented in this paper has the ability to find the correct linking path even when dealing with complex edge structures while avoiding errors such as closed loops of edges that can be noticed in the edge maps returned by the M-SEL algorithm. These results also indicate that our approach deals better with straight edges while the M-SEL algorithm, as illustrated in Figure 11, favors curved edges. As opposed to M-SEL algorithm where a large number of parameters have to be adjusted, our algorithm requires only two parameters to be specified, namely the parameter required to select the threshold parameters of the edge detector and the maximum dimension for the scanning window.

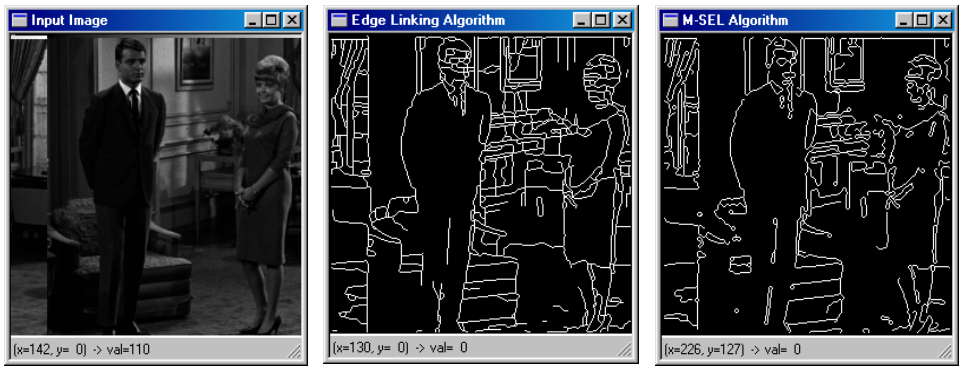

Figure 11. (a) The input image. (b) Edge linking results returned by the proposed algorithm (scanning window 11×11). (c) Edge linking results returned by the M-SEL algorithm.

An important issue is the computational efficiency. Achieving reasonable timing using a complex edge operator such as Canny is difficult, since the computational time required to extract the edge structure derived from a $256 \times 256 \times 256$ greyscale image is 4900 ms when executed on a PC with a Pentium 133 processor. Therefore for this implementation we choose a computationally efficient edge detector, namely the ISEF-based GEF operator where the processing time is 545 ms. Also, it is worth mentioning that this advantage is obtained without significantly reducing the edge recovering performance (see Figure 2). The processing time associated with the linking algorithm depends on the complexity of the edge structure and the size of the scanning window. Timings for images involved in the aforementioned experiments for the proposed algorithm (scanning window 11×11) and M-SEL algorithm are depicted in Table 1.

7. Conclusions

It has generally been believed that enhancing the edge structure has to be based on either computationally intensive multiresolution approaches [6,24] or on methods based on probabilistic relaxation [8]. In this paper we have described a fast and efficient edge linking algorithm to improve the quality of the edge information returned by the ISEF edge detector. The proposed edge linking scheme has two key components. The first component deals with the edge detection and the removal of multiple edge responses.

Table 1

Computational overhead associated with the proposed algorithm and M-SEL algorithm

Input image	Proposed algorithm (sec)	M-SEL algorithm (sec)
Figure 8a	1.32	163
Figure 9a	1.43	165
Figure 10a	1.71	174
Figure 11a	1.57	166

The second component attempts to correct the local imperfections in the edge detected output by maximally exploiting the information around singular points. The resulting algorithm is computationally efficient as it requires a single pass through the entire edge detected output and has the particular advantage that it can be applied to scenes where no *a priori* knowledge is available. Also, experimental results indicate that the performance of the proposed algorithm is comparable to that offered by more complex edge linking algorithms.

REFERENCES

1. D.H. Ballard and C.M. Brown, Computer Vision, Prentice-Hall, (1982).
2. F. Bergholm, "Edge focusing", *IEEE Trans. on Pattern Analysis and Machine Intell. (PAMI)*, 9(6), 726-741 (1987).
3. J.E. Bresenham, "Algorithm for computer control of a digital plotter", *IBM Systems Journal*, 4(1), 25-30 (1965).
4. J. Canny, "A computational approach to edge detection", *IEEE Trans. on Pattern Analysis and Machine Intell. (PAMI)*, 8(6), 679-698 (1986).
5. S. Casadei and S.K. Mitter, "A hierarchical approach to high resolution edge contour reconstruction", in *Proc. of the IEEE Conf. for Computer Vision and Pattern Recognition (CVPR'96)*, San Francisco, USA (1996).
6. A. Farag and E.J. Delp, "Edge linking by sequential search", *Pattern Recognition*,

28(5), 611-633 (1995).

7. A.K. Gupta, S. Chaudhury and G. Parthasarathy, "A new approach for aggregating edge points into edge segments", *Pattern Recognition*, 26(7), 1069-1086 (1993).
8. E.R. Hancock and J. Kittler, "Edge-labelling using dictionary-based relaxation", *IEEE Trans. on Pattern Analysis and Machine Intell. (PAMI)*, 12(2), 165-181 (1990).
9. R. Haralick, "Digital step edges from zero crossing of second derivatives", *IEEE Trans. on Pattern Analysis and Machine Intell. (PAMI)*, 6(1), 58-68 (1984).
10. A. Hajjar and T. Chen, "A VLSI architecture for real-time edge linking", *IEEE Trans. on Pattern Analysis and Machine Intell. (PAMI)*, 21(1), 89-94 (1999).
11. R.M. Haralick and L.G. Shapiro, Computer and robot vision, Addison - Wesley Publishing Company, 33-37 (1992).
12. M. Heath, S. Sarkar, T. Sanocki and K. Bowyer, "Comparison of edge detectors: Initial study and methodology", *Computer Vision and Image Understanding*, 69(1), 38-54 (1998).
13. T. Lindeberg, "On scale selection for differential operators", in *Proc. of 8-th Scandinavian Conf. on Image Analysis*, Tromso, Norway, 857-866 (1993).
14. L.M. Lifshitz and S.M. Pizer, "A multiresolution hierarchical approach to image segmentation based on intensity extrema", *IEEE Trans. on Pattern Analysis and Machine Intell. (PAMI)*, 12(6), 529-541 (1990).
15. D. Marr and E. Hildreth, "Theory of edge detection", in *Proc. of Royal Society*, London, B 207, 187-217 (1980).
16. F. Meyer and S. Beucher, "Morphological segmentation", *Journal of Visual Communication and Image Representation*, vol. 1, 21–46 (1990).
17. P. Perona and J. Malik, "Scale-space and edge detection using anisotropic diffusion", *IEEE Trans. on Pattern Analysis and Machine Intell. (PAMI)*, 12(7), 629-639 (1990).
18. E. Saber, A.M. Tekalp and G. Bozdagi, "Fusion of color and edge information for improved segmentation and edge linking", *Image and Vision Computing*, 15(10), 769-780 (1997).
19. J. Shen and S. Castan, "An optimal linear operator for step edge detection", *CVGIP: Graphical Models Image Processing*, 54(2), 112-133 (1992).

20. W.E. Snyder, R. Groshong, M. Hsiao, K.L. Boone and T. Hudacko, "Closing gaps in edges and surfaces", *Image and Vision Computing*, 10(8), 523-531 (1992).
21. M. Sonka, V. Hlavac and R. Boyle, Image processing, analysis and machine vision, 2-nd edition, PWS - International Thomson Publishing, 579-581 (1998).
22. D. Vernon, Machine vision: Automated visual inspection and robot vision, Prentice-Hall (1991).
23. L. Vincent, "Morphological grayscale reconstruction in image analysis: applications and efficient algorithms", *IEEE Trans. on Pattern Analysis and Machine Intell. (PAMI)*, 2(2), 176-201 (1993).
24. K.L. Vincken, W.J. Niessen and M.A. Viergever, "Blurring strategies for image segmentation using multiscale linking model", in *Proc. of Computer Vision and Pattern Recognition(CVPR)*, San Francisco, USA, 21-26 (1996).
25. M. Xie, "Edge linking by using causal neighborhood window", *Pattern Recognition Letters*, 13(19), 647-656 (1992).

Paul F Whelan received his B.Eng.(Hons) degree from the National Institute for Higher Education Dublin, a M.Eng. degree from the University of Limerick, and his Ph.D. from the University of Wales, Cardiff. During the period 1985-1990 he was employed by Industrial and Scientific Imaging Ltd. and later Westinghouse (WESL), where he was involved in the research and development of industrial vision systems. He was appointed to the School of Electronic Engineering, Dublin City University in 1990 and currently holds the positions of Associate Professor and Director of the Vision Systems Laboratory.

As well as a wide range of scientific publications, Prof. Whelan co-edited *Selected Papers on Industrial Machine Vision Systems* (1994), and was the co-author of *Intelligent Vision Systems for Industry* (1997) and the main author of *Machine Vision Algorithms in Java* (2000). His research interests include applied morphology, texture analysis, machine vision and systems engineering. He is a Senior Member of the IEEE, a Chartered Engineer and a member of the IEE, SPIE and IAPR. He is also a member of a number of machine vision related conference program committees. Prof. Whelan currently serves on the *IEE Irish Centre* committee, as member of the governing board of the *International Association for Pattern Recognition* (IAPR) and as the chairperson of the *Irish Pattern Recognition and Classification Society* (IPRCS).

Ovidiu Ghita received his B.E. and M.E. degrees in Electrical Engineering from Transilvania University, Brasov, Romania. From 1994 through 1996 he was an Assistant Lecturer in the Department of Electrical Engineering at Transilvania University. Since then, he has been a member of the Vision Systems Group at Dublin City University (DCU) during which time he received his Ph.D. for work in the area of robotic vision. Currently, he holds a position of Postdoctoral Research Assistant in the Vision Systems Laboratory at DCU. His current research interests are in the area of range acquisition, shape representation, object recognition and machine learning.