A computationally efficient method for edge thinning and linking using endpoints

Ovidiu Ghita and Paul F. Whelan
Vision Systems Laboratory, School of Electronic Engineering, Dublin City University, Dublin 9, Ireland
{ghitao,paul.whelan}@eeng.dcu.ie

In this paper we propose a fast method for edge thinning and linking that consists of two phases. The first phase involves the integration of the edge structure by aggregating in a hierarchical manner the edge information contained in a relatively small collection of images of different edge densities. The second phase performs edge thinning and linking using the information associated with the endpoints of the aggregated edge results. The particular novelty of this approach lies in the labelling scheme which assigns the directionality of the 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; Sequential edge reconstruction; Endpoints; Edge linking

1. Introduction

The successful detection of edge information in an image is an important precursor to many image processing and analysis operations. Since edges are determined by sharp changes in grey level transitions, their extraction generally entails a two-stage process. Initially the edges are enhanced using partial derivatives, then, the edge detection output is analysed in order to decide whether a particular pixel is an edge or not [3,8]. Although robust edge detection has been a goal of computer vision for many decades, the current range of edge operators fail to correctly recover the entire edge structure associated with a given image. This is due to the presence of image noise and to the small variations in the grey level (or colour) distribution. Thus, the image noise will generate extraneous edges while a small variation of the image intensity distribution will contribute to gaps in edges. As an immediate result, the meaningful regions derived from the image under analysis are not correctly outlined. To address this issue further processing that takes into account the local information revealed in the edge detection output has to be considered.

There are various techniques which address the problem of improving the quality of the edge detection. Approaches that have been used include morphological methods [1,4,15,18], Hough transform [6], probabilistic relaxation techniques [7], multiresolution methods [5,11,19] and the use of additional information such as colour [13]. In general, morphological approaches offer a fast solution and they attempt to maximally exploit the local information, which unfortunately is not always sufficient. In contrast, multiresolution and multiscale methods try 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 paper, we describe a morphological-based algorithm for edge thinning and linking. A general problem related to morphological approaches is the choice of optimal parameters for the edge operator. In this regard, we proposed an efficient method to tackle this problem by aggregating the edge information contained in a stack of images in a hierarchical manner. Then, the endpoints (edge terminators) are detected and labelled by analysing the local edge structure. Finally, the gaps in edges are bridged in agreement with the directionality of the endpoints. An overview of the developed edge linking algorithm is illustrated in Figure 1.

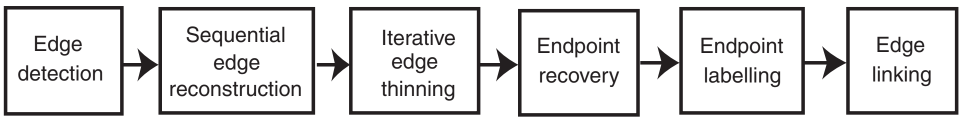

Figure 1. Outline of the edge linking algorithm.

2. Sequential edge reconstruction

As we mentioned earlier, choosing the optimal parameters for an advanced edge operator represents a difficult problem. To reduce the spurious responses generated by image noise, the input image is usually smoothed by applying a Gaussian filter [12]. Consequently the first parameter is the standard deviation σ, a parameter that determines the size and ultimately the scale of the Gaussian operator. To further improve the edge detecting output, Canny [3] proposed a method based on thresholding with hysteresis. This technique evaluates the output of the edge detector using two threshold values (referred to as high and low thresholds) and works as follows: if an edge response is greater than the higher threshold it is considered as a valid edge point. Then, any candidate edge pixels that are connected to valid edge points and are above the lower threshold are also assigned as edge points. A similar approach was employed by Shen and Castan [14] when they developed an optimal edge detector based on ISEF (Infinite Symmetric Exponential Filter) for recovering step-like edges. Because the optimal set of these parameters is dependent on the input image, it is difficult to apply simple criteria to consistently determine these parameters. As most developed systems have been designed to perform a specific task, it makes it difficult to use them in other applications.

To address this problem, many researchers have tried to tackle this issue on a global basis by building a stack of images in which the scale parameter is varied. However, it makes sense to improve the edge structure by aggregating the edge information starting from images with low resolutions towards those with a higher resolution, but this entails a high computational cost since the convolution masks become larger when σ increases. Also choosing the right scales is not a simple issue [10,19]. In addition, the appearance and the localisation of edges within the image are increasingly disturbed when σ increases and this complication may cause a real problem when edges are reconstructed.

To avoid such problems and to maintain a low computational overhead, we choose to vary the threshold parameters while the scale parameter is kept constant to the default value ($\sigma = 1.0$ for Canny or $a_0 = 0.45$ for ISEF-based GEF edge operator [14]). This approach has the advantage that the edge operator has to be applied only once while the thresholding process is sequentially applied to obtain the stack of images with different edge densities.

At this stage, a key problem consists of selecting the optimal range for the threshold parameters. Our aim is to have the edge segments presented in the output image as large as possible. Smaller segments (less than 4 pixels) are generally due to noise. In this regard, we choose the

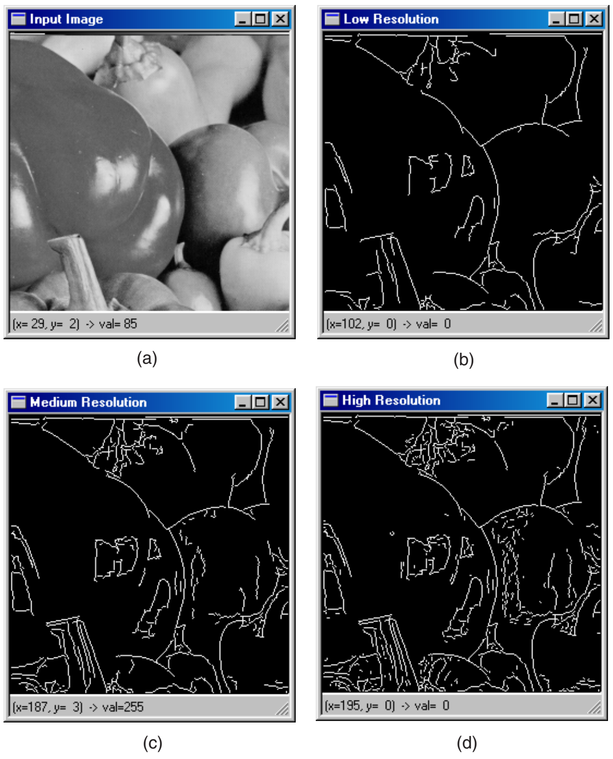

Figure 2. The image stack. (a) Input image. (b) The low edge density image. (c) The medium edge density image. (d) The high edge density image.

lower threshold by analysing the level of small edge segments that are present in the edge detected output. The algorithm increases the value of the lower threshold incrementally (during this phase the higher and lower thresholds are linked together) until the ratio between the number of edge pixels derived from small segments and the number of edge pixels derived from large segments is smaller than a preset value. When this criterion is upheld, the lower threshold value is fixed and by increasing the value of the higher threshold the images with a coarser level of edge detail are obtained. The maximum value for the higher threshold is dependent upon the edge detector used (for example it takes a value of 20 for the GEF edge operator). To increase robustness to noise we considered an image stack which contains three images with a different level of edge detail (see Figure 2).

This solution is advantageous as it allows the removal of noise at each iteration. Also, the scene may contain objects that have a low contrast against the background; consequently they will not be present in the image with the lowest level of edge detail. Once the stack of images are processed (see Figure 2), the algorithm attempts to combine the edge information between them by using the following procedure:

1. Initially the image with lowest edge density is subtracted from the medium edge density image.
2. The resulting image is labelled using a graph-based algorithm [9] and the length of each edge segment is computed.
3. The next step involves analysing the edge structure contained in both images, namely the image with the lowest edge density and the labelled image. If any edge pixel is connected to an edge segment from the labelled image, the edge segment is added to the edge structure in the image with lower edge density.

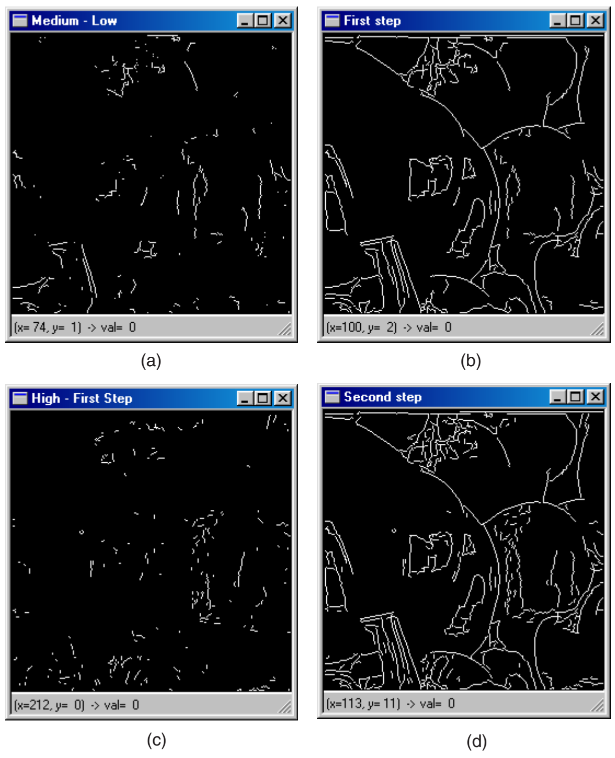

Figure 3. The edge reconstruction process for images illustrated in Figure 2. (a) The subtraction of the low edge density image from the medium edge density image image. (b) The resulting image after first iteration. (c) The subtraction of the resulting image from the high edge density image. (d) The output image.

4. The edge segments from the labelled image that are not connected to the edge structure in the lower edge density image are analysed in order to decide if they are valid edge segments. If the labelled segments under examination contain more than 4 edge pixels, they are added to the edge structure of the image with a lower edge density. The aim of this operation is to remove the isolated and the small undesirable edge responses that are caused by noise.

When this process is completed, the image resulting from the first step is subtracted from the high edge density image. Then, the edges are aggregated using the same procedure outlined above. Figure 3 shows the results obtained after the application of the proposed edge reconstruction scheme.

3. Iterative edge thinning

Multiple edge replications represent another typical error associated with edge detecting operators. Thus, there are cases where the edge responses are several pixels wide. Since our goal consists of reconnecting the interrupted edges using only local information, multiple edge responses may generate incorrect linking decisions. Therefore to use the local information more efficiently, a thinning algorithm has to be applied to remove the unnecessary edge responses. In this regard, an iterative morphological thinning algorithm based on the use of L-type structuring elements was implemented [16]. This algorithm is defined as follows:

$$I \oslash S = I - (I \otimes S) \qquad (1)$$

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. Endpoints 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 bridge the gaps by analysing the singular edge points which are referred to as endpoints. Extracting the endpoints entails a simple morphological analysis [17] and consists of a set of 3×3 masks that are applied to the resultant image after the application of the edge reconstruction process.

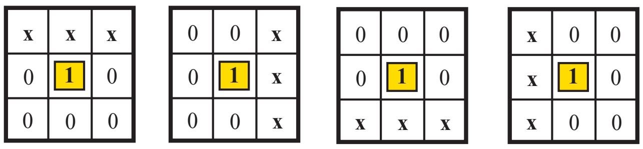

Figure 4. The masks used to detect the endpoints.

Figure 4 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.

To efficiently close the gaps in edges, we need to determine the scanning direction for each endpoint by evaluating the linked edge pixels that generate it.

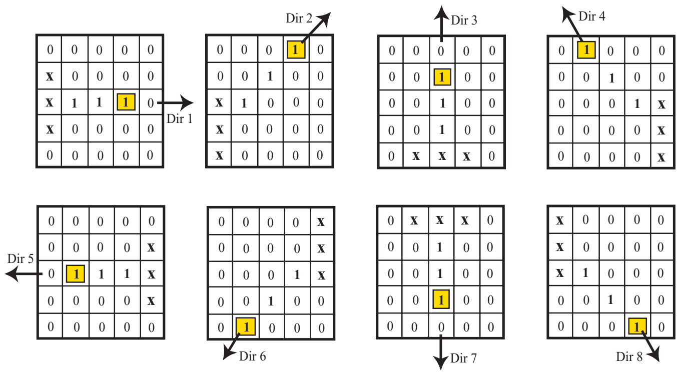

Figure 5. Situations where the edge direction (indicated with an arrow) is derived from straight edges.

As can be easily observed, the masks illustrated in Figure 4 contain some information that gives a useful clue regarding the endpoint direction. Unfortunately, this gives only 4 scanning directions which is not sufficient to always find the correct result. To avoid such limitation we extend the search for edge links to 8 directions, a situation when supplementary information has to be evaluated. As Figure 5 illustrates, there are cases when the endpoint is generated by a straight edge, a situation where the scanning direction can be easily established.

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 6 where the endpoint direction is evaluated by analysing

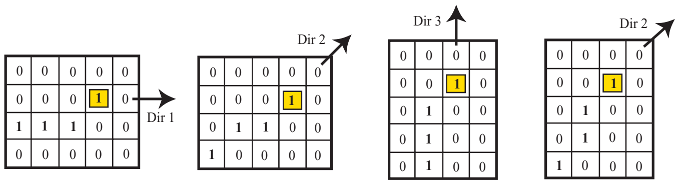

Figure 6. The edge direction derived from curved edges.

the local information for a larger neighbourhood. In Figure 6 only the first 3 directions are illustrated, while the remaining directions can be obtained by rotating the masks.

5. Edge linking

The last step of the algorithm deals with searching for possible edge paths by using the information derived from the endpoints. The scanning process is iterative and starts at the endpoint under investigation.

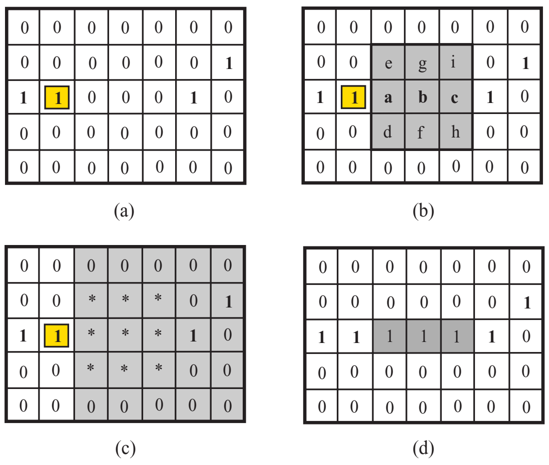

Figure 7. The edge linking process. (a) The edge structure around an endpoint. (b) Scanning the 3 × 3 neighborhood (the pixels are evaluated in alphabetic order). (c) Scanning the 5 × 5 neighborhood (the previous area is not taken into account). (d) The result after the Bresenham algorithm is applied.

This process is defined as follows:

1. Initially the algorithm evaluates the 3 × 3 neighbourhood at the side given by the endpoint direction. In order to avoid closed loops of edges, the pixels situated in the endpoint's neighbourhood are evaluated in a strict order. Thus, the pixels which lie on the endpoint direction are evaluated first. If there are no edge pixels detected, the scanning continues by evaluating the remaining pixels, starting with those closer to the endpoint.
2. If no connections are detected, the algorithm evaluates the 5 × 5 neighbourhood while ignoring the 3 × 3 area which was already assessed.
3. If the scanning process fails to find an edge pixel, the algorithm analyses the 7 × 7 neighbourhood by using the same procedure outlined above.

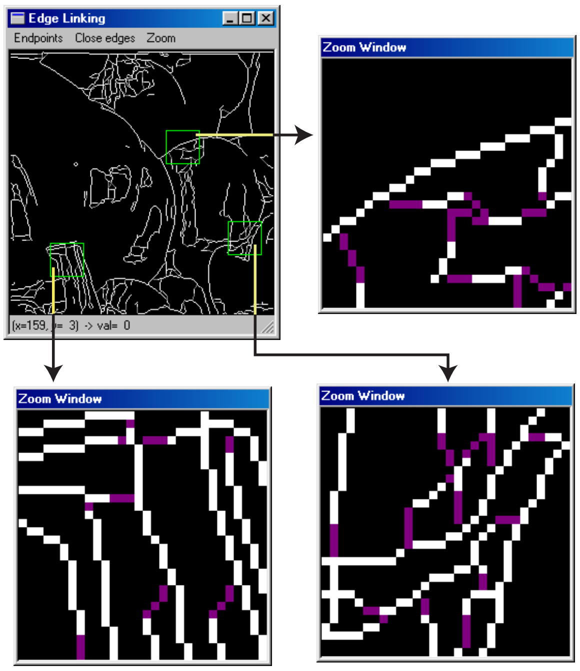

Figure 8. The edge linking results when the algorithm is applied to the image illustrated in Figure 3d. The linking pixels are shaded and for clarity some details are magnified.

4. If a connection is detected, a path is established between the endpoint and the detected edge point by using the Bresenham algorithm [2]. In other words a line is drawn between the endpoint and the edge pixel.

Figure 7 illustrates the edge linking process described above. The mask entries marked with '$*$' indicate that they were already verified.

6. Experiments and results

To evaluate the performance of the proposed edge linking scheme, it was tested on several images. Initially, the algorithm was tested on noiseless images and results of the complete process

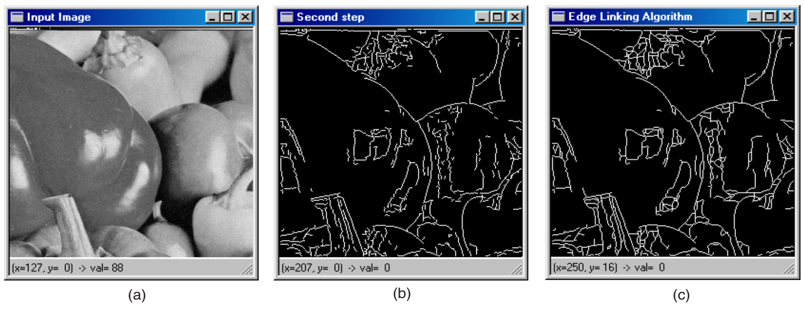

Figure 9. (a) The input image corrupted with Gaussian noise. (b) The result after edge reconstruction. (c) Edge linking results.

can be seen in Figure 8. As Figure 8 illustrates, the algorithm was able to handle even difficult situations such as edge bifurcation. This can be observed in the image details.

To verify the algorithm's robustness to noise, we corrupted the image illustrated in Figure 2a with additive Gaussian noise of standard deviation 30 grey-levels. In Figure 9b the result of the edge reconstruction process is illustrated. Note that the edge segments caused by noise are removed except in the case where they make contact with the edge structure presented in the lower edge density images. Figure 9c illustrates the output after the edge linking algorithm is applied.

Next, to verify the validity of the proposed algorithm, we tested its performance on a range of images defined by various scenes and its results are compared side by side with those returned by the *multiresolution sequential edge linking* algorithm (M-SEL)[5]. Figures 10 and 11 illustrate the results of the proposed edge linking algorithm and the M-SEL algorithm when applied to a set of standard test images.

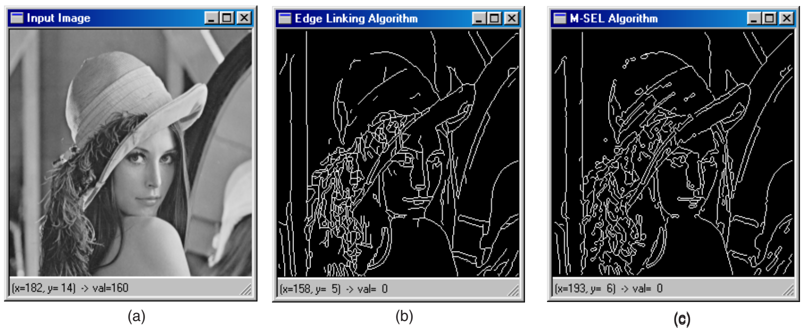

Figure 10. (a) The input image. (b) Edge linking results returned by the proposed algorithm. (c) Edge linking results returned by the M-SEL algorithm.

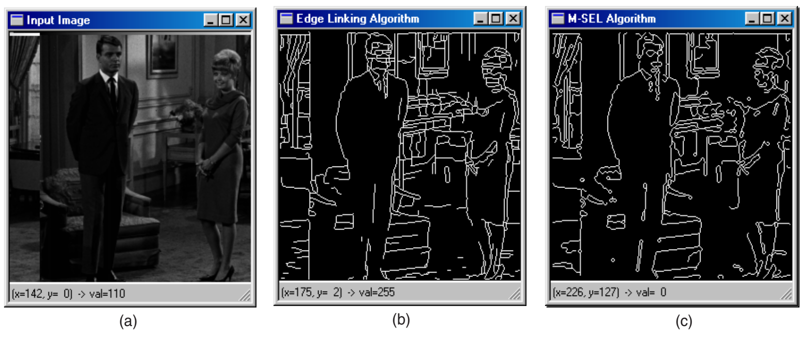

Figure 11. (a) The input image. (b) Edge linking results returned by the proposed algorithm. (c) Edge linking results returned by the M-SEL algorithm.

By analysing the results illustrated above we can conclude that the performance of the proposed algorithm compares well when compared with that offered by the M-SEL algorithm. It can be noticed that our algorithm has the ability to find the correct linking path even in the cases when dealing with complex edge structures while avoiding problems such as closed loops of edges that can be observed in the edge linking 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 can be seen in Figure 11, favors curved edges. At the same time some limitations of our approach can be mentioned. The first is derived from the fact that the proposed algorithm is not able to cope with gaps larger than 7 pixels (M-SEL approach also exhibits the same limitation). The scanning process can be extended to search until a connection is found, but since the gaps are bridged using the Bresenham algorithm [2] the geometry of the curved scene objects is not preserved. However, large gaps cannot be efficiently closed using only the local edge information and to address this problem robustly supplementary knowledge has to be considered.

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 256×256×256 greyscale image is 4900 ms when running on a PC with a Pentium 133 processor (32 Mb RAM and running Windows 98). 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 reducing significantly the edge recovering performance. The processing time associated with the edge reconstruction, thinning and linking algorithm depends on the complexity of the edge structure. Timings for proposed edge linking algorithm and M-SEL algorithm are depicted in Table 1.

Table 1
Computational overhead associated with our algorithm and M-SEL algorithm

Input image	Proposed algorithm (sec)	M-SEL algorithm (sec)
Figure 2a	1.46	163
Figure 9a	1.52	165
Figure 10a	1.58	174
Figure 11a	1.55	166

7. Conclusions

It has been generally believed that edge reconstruction has to be based on either computationally intensive multiresolution approaches [5,19] or on methods which attempt to enhance the edge structure using probabilistic relaxation [7]. In this paper we have described an efficient morphological approach to improve the quality of the edge detection. The proposed edge thinning and linking scheme has two key components. The first component maximises the edge detection *globally* by aggregating the information contained in a small collection of edge images. 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 and has the particular advantage that it can be applied to scenes where no *a priori* knowledge is available. Also, experimental results indicate that in contrast with approaches which involve filtering, the local edge features are not sacrificed at the expense of noise reduction.

REFERENCES

1. D.H. Ballard and C.M. Brown, Computer Vision, Prentice-Hall, (1982).
2. J.E. Bresenham, "Algorithm for computer control of a digital plotter", *IBM Systems Journal*, 4(1), 25-30 (1965).

3. J. Canny, "A computational approach to edge detection", *IEEE Trans. on Pattern Analysis and Machine Intell. (PAMI)*, 8(6), 679-698 (1986).
4. 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).
5. P.H. Eichel and E.J. Delp, "Sequential edge detection in correlated random fields", in *Proc. of the IEEE Computer Vision and Pattern Recognition Conf.*, San Francisco, 14-21 (1985).
6. 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).
7. 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).
8. 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).
9. R.M. Haralick and L.G. Shapiro, Computer and robot vision, Addison - Wesley Publishing Company, 33-37 (1992).
10. T. Lindeberg, "On scale selection for differential operators", in *Proc. of 8-th Scandinavian Conf. on Image Analysis*, Tromso, Norway, 857-866 (1993).
11. 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).
12. D. Marr and E. Hildreth, "Theory of edge detection", in *Proc. of Royal Society*, London, B 207, 187-217 (1980).
13. 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).
14. J. Shen and S. Castan, "An optimal linear operator for step edge detection", *CVGIP: Graphical Models Image Processing*, 54(2), 112-133 (1992).
15. 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).
16. M. Sonka, V. Hlavac and R. Boyle, Image processing, analysis and machine vision, 2-nd edition, PWS - International Thomson Publishing, 579-581 (1998).
17. D. Vernon, Machine vision: Automated visual inspection and robot vision, Prentice-Hall (1991).
18. 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).
19. 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).