Global Journal of Science Frontier Research
Mathematics & Decision Sciences
Volume 12 Issue 4 Version 1.0 April 2012
Type : Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals Inc. (USA)
Online ISSN: 2249-4626 & Print ISSN: 0975-5896

Hypergraph-Based Edge Detection in Gray Images by Suppression of Interior Pixels

By R.Dharmarajan & K.Kannan
SASTRA University

Abstract – This paper presents a new two-stage hypergraph-based algorithm for edge detection in noise-free gray images. The first stage consists of mapping the input image onto a hypergraph called the Intensity Interval Hypergraph (IIHG) associated with the image. In the second stage, each hyperedge is partitioned into two disjoint subsets, namely, the interior pixels and the edge pixels. The interior pixels are then suppressed, so that the edge pixels trace out the edges in the image. These edges are then sharpened using an edge sharpener function to eliminate all the duplicated edges. The algorithm is validated on a number of images of largely varying details, and shows promising results. Other hypergraph-based algorithms are of computational complexity $O(n^2)$ or $O(n^3)$ whereas the IIHG model works at a reduced computational complexity of $O(n)$.

Keywords : *Hypergraph, hyperedge, chessboard metric, interior point, edge.*

AMS subject classification : *05C65, 68U10*

HYPERGRAPH-BASED EDGEDETECTION IN GRAY IMAGES BY SUPPRESSION OF INTERIOR PIXELS

Strictly as per the compliance and regulations of :

Hypergraph-Based Edge Detection in Gray Images by Suppression of Interior Pixels

R.Dharmarajan [α] & K.Kannan [α]

Abstract - This paper presents a new two-stage hypergraph-based algorithm for edge detection in noise-free gray images. The first stage consists of mapping the input image onto a hypergraph called the Intensity Interval Hypergraph (IIHG) associated with the image. In the second stage, each hyperedge is partitioned into two disjoint subsets, namely, the interior pixels and the edge pixels. The interior pixels are then suppressed, so that the edge pixels trace out the edges in the image. These edges are then sharpened using an edge sharpener function to eliminate all the duplicated edges. The algorithm is validated on a number of images of largely varying details, and shows promising results. Other hypergraph-based algorithms are of computational complexity **O (n^2)** or **O (n^3)** whereas the IIHG model works at a reduced computational complexity of **O (n)**.

Keywords / phrases : Hypergraph, hyperedge, chessboard metric, interior point, edge.

I. Introduction

In edge detection, one approach is to track pixels column wise (or, row wise) before using statistical measures for the processing [1]. Graph-based approach [2] identifies binary-related pixels before processing them. Graphs are mathematical modeling tools for low-level image processing applications because graphs are essentially about relationships between objects (these are pixels in images). But graphs do not go beyond binary relations, and pixel relations in images are, in most applications, complex and not necessarily binary. Hence a model that can accommodate higher order relations would be desirable and valuable.

Hypergraphs do precisely that – they accommodate higher order object relations. Hypergraph theory is an original work of Claude Berge [3]. As mathematical entities, hypergraphs are rich and extensive in theory. They also have applications, and published research works [4-7] have shown hypergraphs to be excellent tools in image processing.

The concept of edge is a very familiar one, yet there is no precise rigorous definition of an edge in an arbitrary image. Indeed, the concept as we use it is an abstract one, and so it can give different meaning in different contexts [8]. Several widely accepted ideas of edges and edge detection methods are reported in literature [9, 10]. Essentially, edges in an image correspond to intensity discontinuities or visible intensity changes. The average human eye sees edges in the form of boundaries of objects in the target image. Edge detection, therefore, can be thought of as the process of bringing into view these boundaries while *suppressing* the rest of the image. Broadly, edge detection can be considered a two-stage process: first, the characterization of intensity changes; and second, the use of some structural knowledge to find the edges [11]. Some widely known edge detectors are the Sobel, the Laplacian-of-Gaussian (or LoG) and the Canny edge detectors. However, they do have drawbacks: appearance of undesirable double edges, large and complicated set of rules, and generation of speckles, to mention a few.

Ref.

2. D.J. Morris, M. de J. Lee and A.G. Constantinides, A unified method for segmentation and edge detection using graph theory, ICASSP '86, TOKYO, (1986) 2051-2054.

Author α : Department of Mathematics, SASTRA University, Thanjavur 613402, India. E-mails : claudebergedr@gmail.com, kkannan@maths.sastra.edu

A premise in this paper is that edges are consequences of pixel features and pixel relations. As regards gray images, we have only two aspects at our disposal: the intensity of each pixel and the spatial relationship between pixels. Many algorithms in the traditional class tend to ignore the important spatial relationship aspect [2]. This problem is addressed in the hypergraph framework in this paper.

Wide and thick edges (roughly speaking, these are edges upon edges without any separating features, with one edge following exactly the course of the other) hamper edge detection processes even in clean images by producing undesirable duplication effects in the output image (or, the edge image). So there is a need to characterize not just edges but also duplication of edges to identify the undesirable thick edges before eliminating such. This hypergraph-based work brings some properties of sets and functions into a hypergraph model towards such characterization in a clean image.

The contribution of this article is a novel hypergraph model (called the IIHG, detailed in section 3) for edge detection (in noise-free gray images) with reduced complexity O (n). To the best of our knowledge, the proposed algorithm is the first hypergraph-based one for edge detection with complexity O (n).

The remainder of this paper is organized in sections 2 through 7. Section 2 mentions some published and widely-cited graph- and hypergraph-based edge detection works. Section 3 introduces the hypergraph model that is the base for our algorithm. Section 4 presents the flow of the algorithm in a compact form. Results of experiments on standard test images and real world images are reported and discussed in section 5. This section also features comparative studies to establish the excellent performance and potential of the proposed algorithm. Features of the algorithm are presented in section 6. Concluding remarks form section 7.

II. Related Information on Graph- and Hypergraph-Based Edge Detection

A unified graph-based method for segmentation and edge detection is given in [2], which is in a way a pioneering shift from the traditional approach (of row or column tracking). In [2], mapping of the image onto a graph and computation of shortest spanning trees are important preludes to the process of segmentation and edge detection. However, [2] does not go into the computational complexity of the algorithm . Also, this approach conveys an impression that pixel relations in any image could be simplistic enough to be binary, which impression finds no support in published research.

Bretto and others [4-7] based their research on hypergraph models where patches of pixels (rather than pairs of pixels) are processed by algorithms that are guided principally by the pixel intensity values. This approach is reflective of the fact that hypergraphs are generalizations of graphs. Besides mapping the image onto a hypergraph structure, Bretto et al use the idea of stars and star aggregates. These illustrate possibilities of application of higher order pixel relations in hypergraphs to image processing. But these algorithms are computationally expensive (O (n^3)) and tend to leave unprocessed pixels behind.

III. Hypergraph Representation of a Gray Image

A *hypergraph* is a couple H = (V, E), where V is a nonempty finite set and E is a family of nonempty subsets of V that fills out V. Since E is finite, we index it by a set J = {1, …, k}, k ∈ N, and so we have E = {X_1, . . ., X_k } and $X_1 \cup \ldots \cup X_k$ = V. The set V is called the *vertex set* of H. The family E is called a *hyperedge family* on the vertex set V, and each member of E is called a *hyperedge* (in H).

Ref.

4. Lei Ding and Alper Yilmaz, Interactive image segmentation using probabilistic hypergraphs, Pattern Recognition 43 (2010) 1863-1873.

If the members of E are distinct (meaning: $i \neq j \Longrightarrow Xi \neq X_j$), then H is *simple*. In this case, E is a *set* of nonempty subsets of V. H = (V, E) is called a *partitioned hypergraph* if its hyperedges form a partition of V – i.e., $V = X_1 \cup \ldots \cup X_k$ and $X_i \cap X_j = \varphi$ for $i \neq j$ (where φ denotes the empty set).

To begin with, the input image is represented as a partitioned hypergraph. The hyperedges for this representation are constructed as follows:

Notes

A digital gray image labeled I (and assumed noise-free) is mathematically represented by the function $I: V \rightarrow W$ (where $V \subseteq N \times N$ and W is the set of non-negative integers), where for $a = (x, y) \in V$, I (a) is the gray scale intensity value of the pixel a located at $(x, y) \in N \times N$, so that it is natural to think of the image I as a nonempty finite subset V of N x N. Let V be endowed with the chessboard metric ρ.

Let L be a positive integer, $L \leq 254$ and $q = [255 - 255(\text{mod } L)]/L$. We set

(a) $E_1 = \{a \in V \mid 0 \leq I(a) \leq L\}$,

(b) $E_k = \{a \in V \mid (k-1)L + 1 \leq I(a) \leq kL\}$ for $k = 2, \ldots, q$,

(c) $E_{q+1} = \{a \in V \mid qL + 1 \leq I(a) \leq 255\}$. Obviously the E_t $(t = 1, \ldots, q + 1)$ are subsets of V, some possibly empty (φ).

Let $E = \{E_t \mid t = 1$ through $q + 1$; and $E_t \neq \varphi''$. Then E is a set of nonempty subsets of V, and E fills out V. We take H = (V, E). Then H is a hypergraph on the set V, and thereby is a hypergraph representation of the image I. We call this the *Intensity Interval Hypergraph* (IIHG) associated with the image I. This hypergraph is a partitioned one.

The essential mathematics for the algorithm is given in the appendix (after the references), where all the theory (A1 through A5) is within the framework of the IIHG on V detailed above.

IV. Proposed Algorithm

Figure 1 below gives the flow of the proposed IIHG algorithm. The input image data (box numbered 1 in Fig. 1) are as follows:

1(a) V = set of pixels of the image I (as a finite nonempty subset of $N \times N$)

1(b) Gray scale intensity matrix of V.

1(c) Domain distance metric ρ (Chessboard metric) on V

1(d) Parameter L (called 'intensity interval')

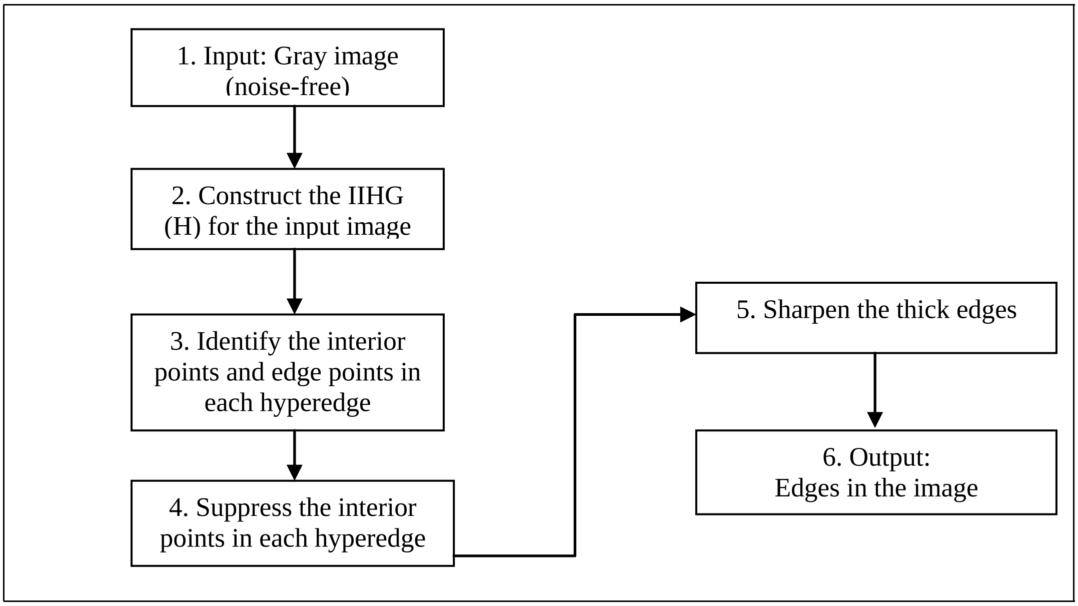


Figure 1 : The IIHG algorithm flow diagram

V. Experiments and Discussion

The computing environment for coding the proposed IIHG algorithm has the following principal components:

(i) Computer category: Micro

(ii) Processor: Intel i13, 3.2 GHz

(iii) Software: MATLAB® 7.0.1

In the proposed algorithm, the output showing the edges depends on the number of hyperedges. The more the number of hyperedges, the denser the edges in the output image.

a) Test reports

Figure 2 below is a simple illustration of how the edge detection algorithm works on a 10 x 10 image patch for L = 90. This patch is a part of a test image from [12].

Ref.

12. http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/BSDS300/html/dataset/images.html

207 P_{1}	158 P_{11}	121 P_{21}	138 P_{31}	131 P_{41}	128 P_{51}	132 P_{61}	133 P_{71}	129 P_{81}	128 P_{91}
127 P_{2}	142 P_{12}	172 P_{22}	193 P_{32}	161 P_{42}	139 P_{52}	130 P_{62}	124 P_{72}	127 P_{82}	132 P_{92}
157 P_{3}	171 P_{13}	169 P_{23}	201 P_{33}	155 P_{43}	128 P_{53}	131 P_{63}	133 P_{73}	129 P_{83}	127 P_{93}
183 P_{4}	157 P_{14}	109 P_{24}	101 P_{34}	75 P_{44}	81 P_{54}	114 P_{64}	128 P_{74}	123 P_{84}	123 P_{94}
112 P_{5}	95 P_{15}	92 P_{25}	72 P_{35}	56 P_{45}	69 P_{55}	102 P_{65}	121 P_{75}	126 P_{85}	125 P_{95}
62 P_{6}	65 P_{16}	87 P_{26}	104 P_{36}	85 P_{46}	75 P_{56}	80 P_{66}	98 P_{76}	125 P_{86}	134 P_{96}
73 P_{7}	67 P_{17}	54 P_{27}	74 P_{37}	86 P_{47}	92 P_{57}	76 P_{67}	66 P_{77}	91 P_{87}	123 P_{97}
60 P_{8}	61 P_{18}	58 P_{28}	59 P_{38}	85 P_{48}	106 P_{58}	90 P_{68}	59 P_{78}	66 P_{88}	101 P_{98}
66 P_{9}	61 P_{19}	65 P_{29}	60 P_{39}	55 P_{49}	60 P_{59}	63 P_{69}	61 P_{79}	78 P_{89}	107 P_{99}
68 P_{10}	61 P_{20}	56 P_{30}	51 P_{40}	61 P_{50}	61 P_{60}	57 P_{70}	59 P_{80}	68 P_{90}	92 P_{100}

Figure 2 : The input image is the patch of size 10 x 10. The hundred pixels in this patch are labeled P_1 through P_{100}. The intensity value of each pixel is just above its label. The pixels filled with black are the edge pixels as identified by the algorithm. The pixels without any filling color are the suppressed pixels – these are either the interior pixels suppressed in the partitioning of the hyperedges or the edge pixels suppressed in the sharpening of thick edges.

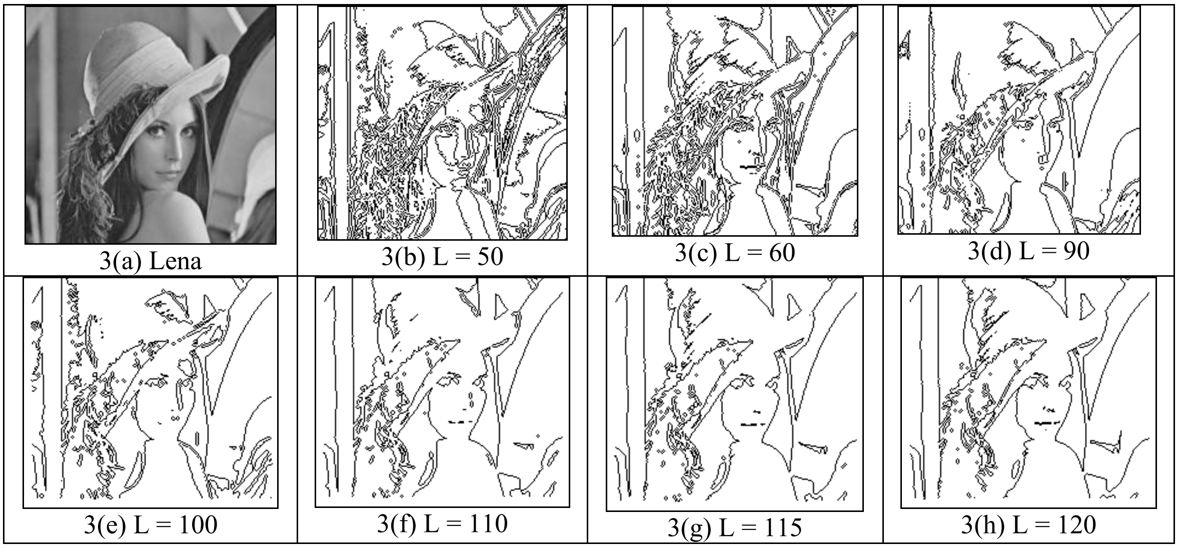
Over six hundred images were taken from [12, 13] and Google Earth which contain ranges of gray images with widely varying features and details. Several of these images appear in published works, and are standard test images – for instance, Lena, Photographer and Peppers – in image processing research. Tests on five widely used images are reported and discussed in this section. The values of the parameter L specified in the reports have been selected after exhaustive testing covering the entire range of L ($1 \leq L \leq 254$). For each image reported here, the selected values of L produce visually more credible results (to the subjective human eye) than its other values. As is always the case in any low level image processing, the judgment of the visual results shown in the examples is subjective.

In fig. 3(b) and 3(c), the outputs show 'cluttering' of edges for $L = 50$ and $L = 60$, respectively. For $L > 80$, the edges become more distinguishable. However, as L is increased, some of the edges may actually disappear- for instance, in the edge image for $L = 100$, the outline of the lips has all but vanished. It is inferred that large values of L could result in loss of edges. And this is in direct contrast to 'too many edges' (or, cluttering) resulting from a low value of L. As regards the Lena image, our inference is that $80 \leq L \leq 95$ is a good range for the edge image to be a reliable representative of the true edges in the original (to the human eye). A similar inference can be made for each image tested.

Ref.

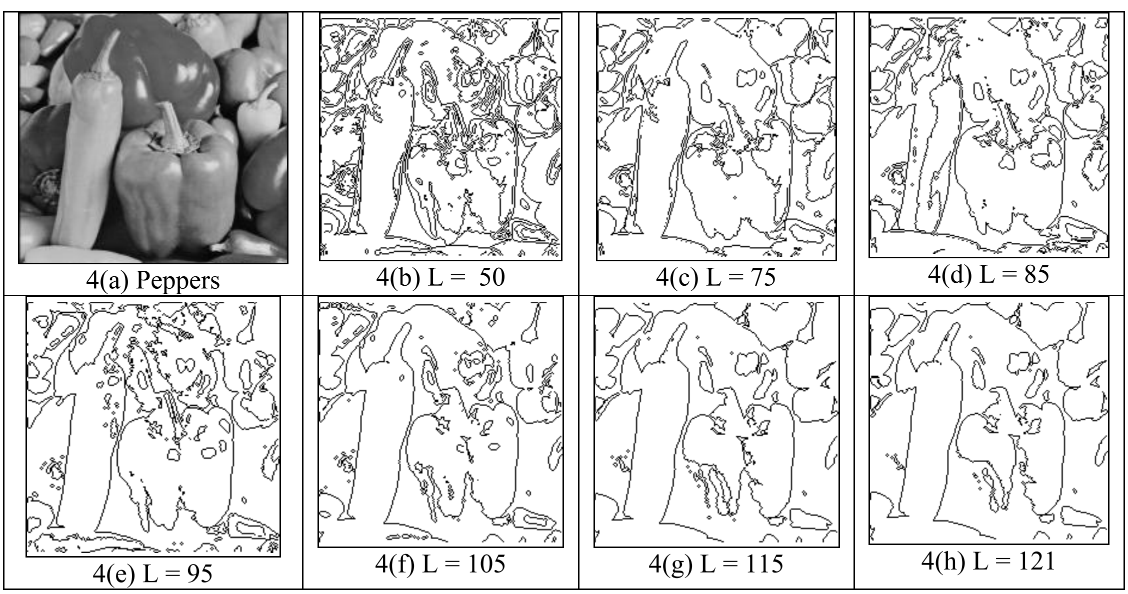
13. http://www.vision.caltech.edu/htmlfiles/archive.html

In each of the following test reports, the first image (a) is the input (original). The others are the output edge images for the specified values of L.



3(a) Lena | 3(b) L = 50 | 3(c) L = 60 | 3(d) L = 90
3(e) L = 100 | 3(f) L = 110 | 3(g) L = 115 | 3(h) L = 120

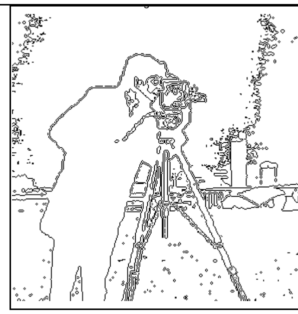
Another widely used test image – Peppers – features in figure 4. The range 90 ≤ L ≤ 120 gives better edge representations. The Lena and the Peppers images also feature in the comparison of our proposed algorithm with three other edge detection algorithms, shown in section 5.2

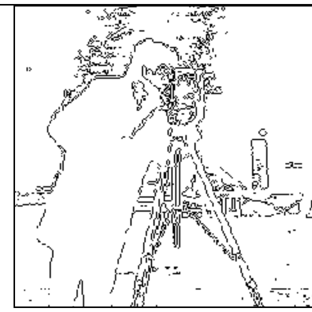


4(a) Peppers | 4(b) L = 50 | 4(c) L = 75 | 4(d) L = 85
4(e) L = 95 | 4(f) L = 105 | 4(g) L = 115 | 4(h) L = 121

Notes

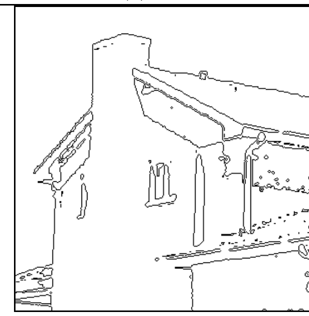

5(a)Photographer


5(b) L = 80

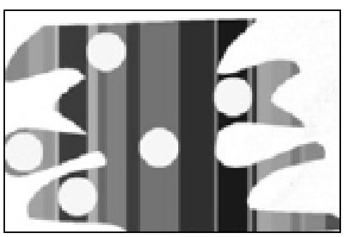

5(c) L = 90

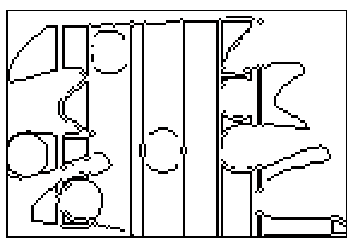

6(a) House

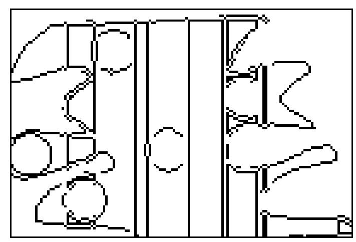

6(b) L = 70

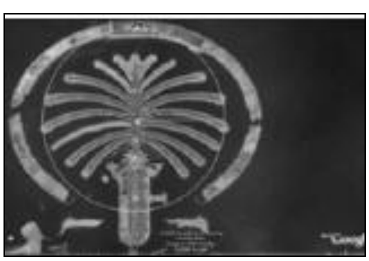

6(c) L = 90

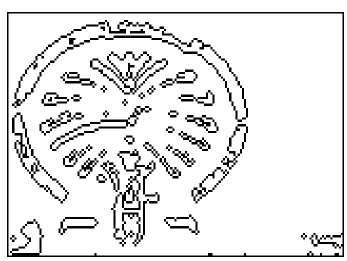
Synthetic and real world images were tested with a view to stress-testing the code. Results on two such images are seen in figures 7 and 8 below.

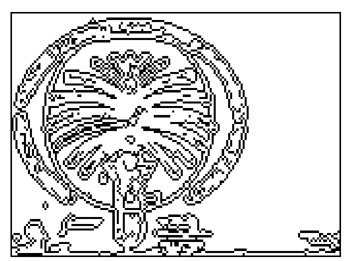

7(a) Synthetic image (Syn 1)


7(b) Edges in Syn 1 (L = 90)


7(c) Edges in Syn 1 (L = 100)


8(a) Real world image (RW1)
(Acquired from Google Earth)


8(b) Edges in RW 1 (L = 128)


8(c) Edges in RW 1 (L = 65)

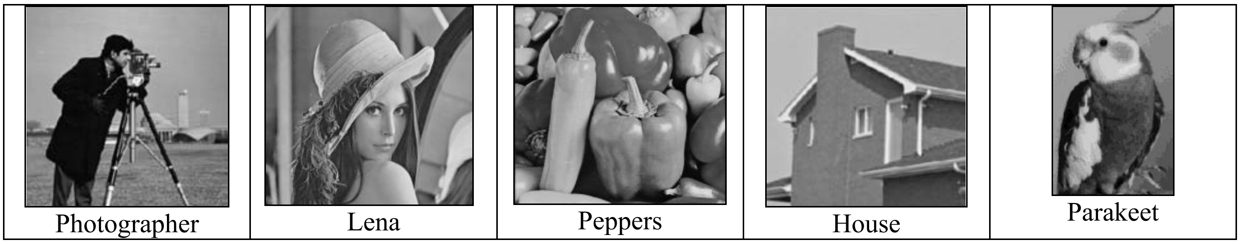
The images reported here are of different sizes (80 x 80 to 512 x 512) and detail contents. CPU run time is more for some of these images because of their larger size.

b) Comparisons and performance reports

The proposed IIHG algorithm was compared for edge detection results with three other published algorithms – namely, the Sobel [14], the Canny [15], and the MG-IT2FIS [16]. The original images (inputs) are in panel 1. The results of each of the four algorithms on these five images are in panel 2. Figures in these panels have not been numbered. As can be seen from panel 2, the comparison works out, to a significant extent, in favor of the proposed IIHG-based algorithm.

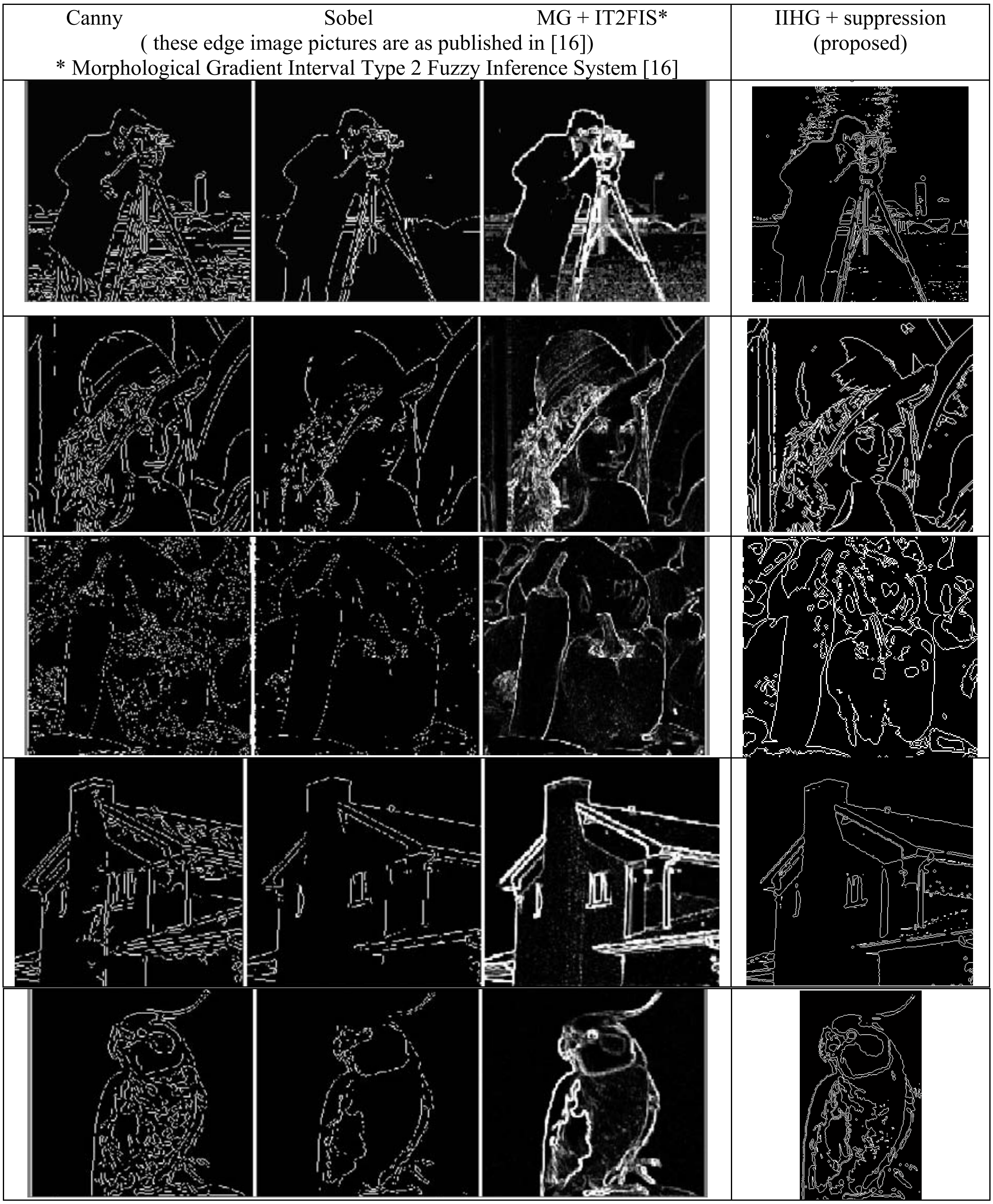
Ref.

15. John Canny, A computational approach to edge detection, IEEE transactions on Pattern Analysis and Machine Intelligence, 6 (1986) 679-698.



Panel 1 : The five original images that feature in the comparison (shown in panel 2)

Notes



Panel 2 : Comparison of the proposed IIHG algorithm with Sobel, Canny and MG+IT2

Notes

In the above comparison experiment, we used L = 90 (for the Photographer image), 80 (Lena), 95 (Peppers), 90 (House) and 60 (Parakeet).

Table 1 : Performance of the proposed algorithm (Standard test images) #H: Number of hyperedges; *Run time in seconds, rounded to one place after the decimal

S.no.	Image	L	#H	Run time*	S.no.	Image	L	#H	Run time*
1	Lena (200 x 200; Bitmap)	50	6	13.8	4	Peppers (200 x 200; JPEG)	50	6	12.8
		60	5	12.7			75	4	10.5
		90	3	10.3			85	3	10.9
		100	3	11.2			95	3	9.5
		110	3	11			105	3	8.8
		115	3	10			115	3	8.4
2	Photographer (333 x 336;Bitmap)	80	4	15.8	5	House (333 x 333; Bitmap)	70	4	10
		90	3	17.4			90	3	8.1
		105	3	17.9			100	3	7
3	Parakeet (328 x 198; Bitmap)	60	5	15	6	Other images** TIF / JPEG / Bitmap	40 to 150	2 to 7	3 to 40
		70	4	13.2					
		85	3	12.4					

** More than 350 images from [12] and [13] (size: 80 x 80 to 512 x 512)

Table 2 : Performance of the proposed algorithm (Synthetic and real world images)

S.no.	Image & size	L	#H	Run time*
1	Syn 1 (94 x 150; Bitmap)	90	3	8.2
		100	3	8.6
		55	5	7.1
2	RW1 (108 x 168; Bitmap)	128	2	10.5
		93	3	14
		65	4	16.6
3	Other images*** TIF / JPEG / Bitmap	40 to 150	2 to 7	6 to 40

*** More than 300 images from [12] and [13] and Google Earth (size: 90 x 90 to 512 x 512)

c) Computational complexity of the proposed algorithm

The number of hyperedges in the first stage does not exceed q + 1, where $q = [255 - 255 \pmod{L}] / L$. So for any positive integer value of L, we have $q \le 255$. Since the hyperedges are non-intersecting, each pixel is visited exactly once in the first stage.

In the second stage, in each hyperedge, each pixel is visited at most four times for segregating the edge points from the interior points. Then, to suppress each interior pixel, we need exactly one assignment of the value 255 to the pixel. Subsequently, to identify the thick edges, each edge pixel is visited at most four times. And the sharpening process that follows takes one assignment operation (the one mentioned above) for every pair of thick edge pixels that correspond in the required bijective way (see A3 of appendix). Hence the number of computations in the algorithm is λn, with $1 \le \lambda \le 10$, where n = number of pixels in the input image. Let $f(n) = n$ and $g(n) = \lambda n$. As n tends to ∞, the limit of $f(n) / g(n)$ is $1 / \lambda$, and that of $g(n)/ f(n)$ is λ. Since both λ and $1/ \lambda$ are finite and nonzero, we aver that the complexity of the proposed algorithm is O(n).

VI. Algorithm Features

(i) The sequential combination of two functions – thick edge identifier and thick edge sharpener, in that order – ensures that no redundancies appear in any edge. This combination is effective principally because of the IIHG model.

(ii) The first stage (construction of the IIHG) ends when the empty set (φ) takes the place of V, and this happens in at most q + 1 steps. The second stage (edge detection) ends when each hyperedge has been cleared of its thick edges, which happens in at most S steps (but in most images well below S because of interior points being excluded from this process), where $S = \Sigma\,\Sigma\,(|E_j| \times |E_k|)$, the sums running over the indices j (second) and k (first) with $j, k \in \{1, \ldots, |E|\}$ and $j < k$; and whatever the value of L, $|E|$ does not exceed 255. Thus the algorithm is convergent.

(iii) The algorithm handles large sized images of varying dimensions and for all values of L in its stipulated range ($1 \leq L \leq 254$), and so is robust.

(iv) The algorithm is fast for test images that are widely used as standards by researchers in image processing (for instance, Lena and Peppers).

(v) Since the output is always viewed rather subjectively, edges that are considered 'not desirable' can be removed by tuning L. While this is a facility that is in-built in the algorithm, tuning L to eliminate such 'undesirable' edges could accidentally rub out true edges also. This is one limitation of the algorithm.

VII. Concluding Remarks

(i) We have presented a hypergraph-based one-parameter-driven partitioning algorithm for edge detection in clean gray images. The algorithm processes patches of pixels of arbitrary (finite) size and distribution efficiently. The computational complexity is O(n), which is an outstanding feature here.

(ii) From the tests reported in section 5.1, we have arrived at an apparently good range for L for a large number of images, standard or real-time, and this is $60 \leq L \leq 120$. However, L is image-dependent. Going by our tests (on hundreds of standard, real-time and synthetic images), we report that $L < 60$ tends to clutter the output figure with too many edges because false edges are shown among the true ones. On the other hand, for $L > 120$ could accidentally suppress considerable number of edge pixels, resulting in loss of true edges. Since performance of parameter-driven algorithms are application-dependent, we have not gone into the question of optimizing L.

(iii) In image engineering applications, the input image may have to be first subjected to a noise removal scheme before the IIHG algorithm is applied. As for noise removal, adequate schemes are available [5, 6, 17-20].

VIII. Acknowledgements

The authors thank Prof. R. Sethuraman, Vice-Chancellor, SASTRA University, Thanjavur, for his invaluable encouragement and support. The authors also express their thanks to: Dr. R. Balakrishnan, DST-Ramanujan Research Chair, SASTRA University and Dr. A. K. Singh, Scientist-G, DST, Government of India, for their valuable suggestions and encouragements. (DST is the Department of Science and Technology.)

Ref.

16. Patricia Melin, Olivia Mendoza, Oscar Castillo, An improved method for edge detection based on interval type-2 fuzzy logic, Expert Systems with Applications, 37 (2010) 8527–8535.

Notes

References Références Referencias

1. R.C. Gonzalez, R.E. Woods, Digital Image Processing, Pearson Education, Singapore, 2005.
2. D.J. Morris, M. de J. Lee and A.G. Constantinides, A unified method for segmentation and edge detection using graph theory, ICASSP '86, TOKYO, (1986) 2051-2054.
3. Claude Berge, Hypergraphs: Combiantorics of finite sets, North-Holland, Amsterdam, 1989.
4. Lei Ding and Alper Yilmaz, Interactive image segmentation using probabilistic hypergraphs, Pattern Recognition 43 (2010) 1863-1873.
5. Alain Bretto and Luc Gillibert, Hypergraph-based image representation, GbRPR 2005, LNCS 3434 (2005) 1-11.
6. Alain Bretto, Hocaine Cherifi and Driss Aboutajdine, Hypergraph imaging: An overview, Pattern Recognition 35 (2002) 651-658.
7. A. Bretto, J. Azema, H. Cherifi and B. Laget, Combinatorics and image processing, Graphical Models and Image Processing, Vol. 59, No. 5 (1997) 265-277.
8. Ting-jen Yen, A Qualitative profile-based approach to edge detection, Ph. D. thesis (2003), New York University.
9. Ehsan Nadernejad, Sara Sharifzadeh and Hamid Hassanpour, Edge detection techniques: evaluations and comparisons, Applied Mathematical Sciences, Vol. 2, No. 31 (2008) 1507 – 1520.
10. Tamar Peli and David Malah, A study of edge detection algorithms, Computer Graphics and Image Processing, 20 (1982) 1-21.
11. S. Sitharama Iyengar and Weian Deng, An efficient edge detection algorithm using relaxation labeling technique, Pattern Recognition, Vol. 28, No. 4 (1995) 519-536.
12. http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/BSDS300/html/dataset/images.html
13. http://www.vision.caltech.edu/htmlfiles/archive.html
14. J Kittler, On the accuracy of the Sobel edge detector, Image and Vision Computing, Vol. 1, no. 1 (1983) 37-42.
15. John Canny, A computational approach to edge detection, IEEE transactions on Pattern Analysis and Machine Intelligence, 6 (1986) 679-698.
16. Patricia Melin, Olivia Mendoza, Oscar Castillo, An improved method for edge detection based on interval type-2 fuzzy logic, Expert Systems with Applications, 37 (2010) 8527–8535.
17. S. Rital, A.Bretto, H.Cherifi and D.Aboutajdine, A combinational edge detection algorithm on noisy images, International Symposium on Video Image Processing and Multimedia Communications (2002) 351-355.
18. S. Rital, A.Bretto, H.Cherifi and D.Aboutajdine, Application of adaptive hypergraph model to impulsive noise detection, Lecture Notes in Computer Sciences, Springer-Verlag (2001) 34-42.
19. K. Kannan, B. Rajesh Kanna and C. Aravindan, Root mean square filter for noisy images based on hyper graph model, Image and Vision Computing, Vol. 28, Issue 9 (2010) 1329-1338.
20. R.Dharmarajan and K.Kannan, A hypergraph-based algorithm for image restoration from salt and pepper noise, AEU-International Journal of Electronics and Communications Volume 64, Issue 12, (2010) 1114-1122.

APPENDIX

Bonded sets

Let $N \times N$ denote the Cartesian square of the set N of positive integers. For $(x_1, y_1), (x_2, y_2) \in N \times N$, we define $\rho((x_1, y_1), (x_2, y_2)) = \max\{|x_1 - x_2|, |y_1 - y_2|\}$. The function ρ is a metric on N x N – and hence on any nonempty subset of N x N – and ρ is called the *chessboard metric*.

For a given nonempty set V, by 2^V we mean the power set of V; and by 2^{V*} we mean the set of all nonempty subsets of V. Let V be a finite nonempty subset of N x N endowed with the chessboard metric.

If $X \in 2^{V*}$ and $a \in V$, we define $\rho(a, X) = \min\{\rho(a, b): b \in X$". If $X, Y \in 2^{V*}$ then we define $\rho(X, Y) = \min\{\rho(a, b): a \in X, b \in Y$".

Let A be a nonempty subset of V. A finite sequence $x_1, \ldots, x_k$ of elements of A is called a *1-step sequence* (1-ss) in A if $\rho(x_i, x_{i+1}) = 1$ for each $i = 1, \ldots, k-1$. If a, b $\in$ A, then we say a is *bonded* to b in A if $\rho(a, b) \leq 1$ or if there exist points $z_1, \ldots, z_k$ in A such that the sequence $a, z_1, \ldots, z_k, b$ is a 1-ss in A. In this case we write –a: b"A.

Clearly: (i) {a: a"A, (ii) {a: b"A ⟹ {b: a"A and (iii) {a: b"a and {b: c"A ⟹{a: c}A, for all a, b, c $\in$ A. Further, {a: b"A ⟹ {a: b"B whenever $A \subseteq B$. A is called a *bonded set* if {a: b"A for every a, b $\in$ A. A singleton set is obviously bonded.

Interior points and edge points in an image

Given $a = (x, y) \in V$, we define the neighborhood $B_4(a)$ as:

$B_4(a) = \{b = (p, q) \in V \mid \rho(a, b) \leq 1$ and (x = p or y = q)". Clearly $a \in B_4(a)$ for each $a \in V$.

Let $A \in 2^{V*}$ and $a = (x, y) \in A$. We say a is an *interior point* of A if and only if $B_4(a) \subseteq A$. We let IntA denote the set of all the interior points of a given set A. If a is not an interior point of A then we call it an *edge point* of A.

Edges in an image

By $|A|$ we mean the cardinality (or, size) of the set A. An *edge* in V is a nonempty subset e (V) of V with the following properties:

(ed-1) $e(V) \subseteq X$ for some (hence unique) hyperedge X in H;

(ed-2) $|e(V)| > 1$;

(ed-3) no point of e (V) is an interior point of X;

(ed-4) e (V) is bonded, and

(ed-5) if Y satisfies (i) $e(V) \subset Y \subseteq X$, (ii) $e(V) \neq Y$ and (iii) $Y \cap \mathrm{Int}X = \varphi$, then Y is not bonded.

A *thick edge* (or, a duplicated edge) in V is a nonempty subset r(V) of V that can be partitioned as $r(V) = r_1(V) \cup r_2(V)$ (i.e., $r_1(V)$ and $r_2(V)$ are nonempty subsets of r(V) such that $r_1(V) \cap r_2(V) = \varphi$) with the following properties:

(t-ed-1) $|r_1(V)| = |r_2(V)|$,

Notes

(t-ed-2) $r_1(V) \subseteq X_1$ and $r_2(V) \subseteq X_2$ for some distinct (hence disjoint) hyperedges X_1 and X_2 in H (we call X_1 and X_2 the *source hyperedges* of $r_1(V)$ and $r_2(V)$, respectively), and

(t-ed-3) there exists a bijective map $f: r_1(V) \to r_2(V)$ such that for each $a \in r_1(V)$ we have $a \in B_4(f(a))$ as well as $f(a) \in B_4(a)$.

Notes

Suppression of interior points

Let $A \in 2^{V^*}$ and $b^*(A) = A - \mathrm{Int}A$, where IntA denotes the set of all the interior points of A. Let. Evidently $b^*(A)$ is nonempty unless $A = V$. We call the computation of $b^*(A)$ the *suppression* of the interior points of A. Notice that if $A \in E$, then $b^*(A)$ is either an edge in V or a union of edges in V.

Sharpening of thick edges

Given two distinct hyperedges X_1 and X_2 in H, we write $X_1 < X_2$ if $I(a) < I(b)$ for every $a \in X_1$ and $b \in X_2$. Let $r(V) = r_1(V) \cup r_2(V)$ be a thick edge in V with source hyperedges X_1 and X_2, respectively, such that $X_1 < X_2$. Let $\psi: r_2(V) \to W$ be the constant function $\psi(b) = 255$. The function ψ is called the edge sharpener function. It suppresses one half of the targeted thick edge out of the picture, so that only the other half is seen in the edge image.

Notes

This page is intentionally left blank