International Journal of Image and Graphics
Vol. 15, No. 1 (2015) 1550003 (21 pages)
© World Scientific Publishing Company
DOI: 10.1142/S0219467815500035

3D Edge Detection Based on Boolean Functions and Local Operators

Ricardo Dutra da Silva
Department of Informatics, Federal University of Technology, Paraná
Curitiba-PR, Brazil 80230-901
rdutra@dainf.ct.utfpr.edu.br

Rosane Minghim
Instituto de Ciências Matemáticas e de Computação
University of São Paulo, São Carlos-SP, Brazil 13560-970
rminghim@icmc.usp.br

Helio Pedrini
Institute of Computing, University of Campinas
Campinas-SP, Brazil 13083-852
helio@ic.unicamp.br

Received 5 December 2013
Accepted 24 November 2014
Published 23 February 2015

Edge detection is one of the most commonly used operations in image processing and computer vision areas. Edges correspond to the boundaries between regions in an image, which are useful for object segmentation and recognition tasks. This work presents a novel method for 3D edge detection based on Boolean functions and local operators, which is an extension of the 2D edge detector introduced by Vemis *et al.* [*Signal Processing* **45**(2), 161–172 (1995)] The proposed method is composed of two main steps. An adaptive binarization process is initially applied to blocks of the image and the resulting binary map is processed with a set of Boolean functions to identify edge points within the blocks. A global threshold, calculated to estimate image intensity variation, is then used to reduce false edges in the image blocks. The proposed method is compared to other 3D gradient filters: Canny, Monga–Deriche, Zucker–Hummel and Sobel operators. Experimental results demonstrate the effectiveness of the proposed technique when applied to several 3D synthetic and real data sets.

Keywords: Edge detection; Boolean functions; 3D data sets.

1. Introduction

Edges are considered as an important feature for image understanding and scene analysis. The detection of abrupt changes or discontinuities captures relevant properties of the image. Ideally, the result of applying an edge detector to a digital 2D image is a set of connected curves that indicate the boundaries of objects or regions.

Since edges contain basic structural properties of the image, edge detection methods may significantly reduce the amount of data to be processed in subsequent tasks of interpreting the information contents in the original image.[1]

Unfortunately, it is generally difficult to extract edges from images. In many cases, edges suffer from fragmentation, that is, the edge curves are not always connected. Furthermore, the extraction process may result in edge displacement or false edges.

In many applications, from areas such as medicine and industry, images are represented in three dimensions, usually a volume of data formed by multiple 2D cross-sectional images. The problem of edge detection in images can be summarized as the identification of discontinuities of a discrete function, which are lines in 2D[2,3] and surfaces in 3D.[4,5] Edge detection techniques have been extensively investigated for 2D images, however, the problem has not yet received much attention in 3D.

The method presented in this paper is an extension to three dimensions of the edge operator proposed by Vemis *et al.*,[6] which uses Boolean functions and local operations to detect edges in 2D images. To demonstrate the effectiveness of the method, its results are compared to two commonly employed 3D gradient filters.

The text is organized as follows. Section 2 gives a short discussion on the edge detection problem. Section 3 describes the proposed method. Experimental results are presented and discussed in Sec. 4. Conclusions obtained in this work are presented in Sec. 5.

2. Related Work

Several image edge detection approaches have been described in the last few decades. Extensive surveys on the subject can be found in the literature.[7–10] The majority of the methods are used to detect edges in 2D images.

Most edge detection algorithms for 2D images are based on either maximum in the first derivative or zero crossings in the second derivative of the image. Gradient magnitude and direction are commonly used as a first-order derivative to compute a measure of edge strength. Some known edge operators, such as Roberts,[11] Prewitt[12] and Sobel,[13] convolve the original image with a set of masks to obtain an approximation of the image intensity gradient.

Zero crossings in a second-order derivative can be computed from the Laplacian operator or a nonlinear differential expression. A smoothing stage can be applied to the image to reduce noise effect. Marr–Hildreth edge detector[14] is based on the search of zero-crossings of the Laplacian operator applied to a Gaussian-smoothed image. The method, however, suffers from some limitations. It can produce false edges and give poor localization at curved edges.

Canny[15] derived an edge detector that uses a filter based on the first derivative of a Gaussian to generate a slightly blurred version of the original image. Then, gradient information is used to define edge points in the image through a process known as non-maximum suppression to produce thin edge curves. Edge tracing depends on thresholds at a given intensity gradient. Canny operator uses thresholding with hysteresis to trace paths of the edge through the image and produce continuous curves.

Canny–Deriche detector[16] was proposed from similar mathematical criteria as the Canny edge detector, resulting in a set of recursive filters for image smoothing instead of exponential filters or Gaussian filters. The recursive filters come from the optimal filter constructed by Deriche

$$f(x) = -cxe^{-\alpha|x|}. \tag{1}$$

Monga and Deriche[17] extended the Canny–Deriche method to three dimensions, presenting the computation of the recursive filters.

Vemis *et al.*[6] presented a 2D edge detection method based on local operations and Boolean functions. A local threshold is used to transform the multilevel image into a binary one. The existence of edge points is verified through a local threshold and nonlinear filtering using Boolean functions.

Liu[18] proposed a first generalization in three dimensions of the Roberts edge operator using a symmetric gradient operator. Morgenthaler and Rosenfeld[19] formulated an edge detector by fitting quadric surfaces to a neighborhood of each image point and by taking the magnitude of the surface gradient as an estimate of the change rate of gray level at that point. Luo *et al.*[20] proposed a 3D edge operator using moments. Partial derivatives have been used in a number of operators to find edges in 3D images.[17,21–24]

Zucker and Hummel[25,26] proposed an extension to 3D of the Hueckel operator.[27] An edge is considered as a plane passing through the center of a unit volume. The purpose is to define an operator that finds the best oriented plane at each point in the image. Their method approximates the operator with the following three basis functions

$$\begin{aligned}\phi_1(x,y,z) &= \frac{x}{\sqrt{x^2+y^2+z^2}},\\ \phi_2(x,y,z) &= \frac{y}{\sqrt{x^2+y^2+z^2}},\\ \phi_3(x,y,z) &= \frac{z}{\sqrt{x^2+y^2+z^2}},\end{aligned} \tag{2}$$

where ϕ_1, ϕ_2 and ϕ_3 correspond to x, y and z directions, respectively. Approximations of the derivatives, where the unit sphere is partitioned into a $3 \times 3 \times 3$ unit cube, are shown in Fig. 1.

The 3D Sobel gradient operator applies $3 \times 3 \times 3$ masks to compute discrete approximations of the partial derivatives of the image intensity. Edge masks for x, y and z directions are shown in Fig. 2.

3. Proposed Method

The proposed method is an extension to three dimensions of the edge detection technique described by Vemis *et al.*[6] Furthermore, unlike the original 2D technique,

$$x\text{-direction:}\quad \underbrace{\begin{array}{|c|c|c|}\hline -\frac{\sqrt{3}}{3} & -\frac{\sqrt{2}}{2} & -\frac{\sqrt{3}}{3} \\ \hline -\frac{\sqrt{2}}{2} & -1 & -\frac{\sqrt{2}}{2} \\ \hline -\frac{\sqrt{3}}{3} & -\frac{\sqrt{2}}{2} & -\frac{\sqrt{3}}{3} \\ \hline \end{array}}_{x-1} \quad \underbrace{\begin{array}{|c|c|c|}\hline 0 & 0 & 0 \\ \hline 0 & 0 & 0 \\ \hline 0 & 0 & 0 \\ \hline \end{array}}_{x} \quad \underbrace{\begin{array}{|c|c|c|}\hline \frac{\sqrt{3}}{3} & \frac{\sqrt{2}}{2} & \frac{\sqrt{3}}{3} \\ \hline \frac{\sqrt{2}}{2} & 1 & \tfrac{\sqrt{2}}{2} \\ \hline \frac{\sqrt{3}}{3} & \frac{\sqrt{2}}{2} & \frac{\sqrt{3}}{3} \\ \hline \end{array}}_{x+1}$$

$$y\text{-direction:}\quad \underbrace{\begin{array}{|c|c|c|}\hline \frac{\sqrt{3}}{3} & \frac{\sqrt{2}}{2} & \frac{\sqrt{3}}{3} \\ \hline 0 & 0 & 0 \\ \hline -\frac{\sqrt{3}}{3} & -\frac{\sqrt{2}}{2} & -\frac{\sqrt{3}}{3} \\ \hline \end{array}}_{y-1} \quad \underbrace{\begin{array}{|c|c|c|}\hline \frac{\sqrt{2}}{2} & 1 & \frac{\sqrt{2}}{2} \\ \hline 0 & 0 & 0 \\ \hline -\frac{\sqrt{2}}{2} & -1 & -\frac{\sqrt{2}}{2} \\ \hline \end{array}}_{y} \quad \underbrace{\begin{array}{|c|c|c|}\hline \frac{\sqrt{3}}{3} & \frac{\sqrt{2}}{2} & \frac{\sqrt{3}}{3} \\ \hline 0 & 0 & 0 \\ \hline -\frac{\sqrt{3}}{3} & -\frac{\sqrt{2}}{2} & -\frac{\sqrt{3}}{3} \\ \hline \end{array}}_{y+1}$$

$$z\text{-direction:}\quad \underbrace{\begin{array}{|c|c|c|}\hline -\frac{\sqrt{3}}{3} & 0 & \frac{\sqrt{3}}{3} \\ \hline -\frac{\sqrt{2}}{2} & 0 & \frac{\sqrt{2}}{2} \\ \hline -\frac{\sqrt{3}}{3} & 0 & \frac{\sqrt{3}}{3} \\ \hline \end{array}}_{z-1} \quad \underbrace{\begin{array}{|c|c|c|}\hline -\frac{\sqrt{2}}{2} & 0 & \frac{\sqrt{2}}{2} \\ \hline -1 & 0 & 1 \\ \hline -\frac{\sqrt{2}}{2} & 0 & \frac{\sqrt{2}}{2} \\ \hline \end{array}}_{z} \quad \underbrace{\begin{array}{|c|c|c|}\hline -\frac{\sqrt{3}}{3} & 0 & \frac{\sqrt{3}}{3} \\ \hline -\frac{\sqrt{2}}{2} & 0 & \frac{\sqrt{2}}{2} \\ \hline -\frac{\sqrt{3}}{3} & 0 & \frac{\sqrt{3}}{3} \\ \hline \end{array}}_{z+1}$$

Fig. 1. $3 \times 3 \times 3$ Zucker–Hummel masks.

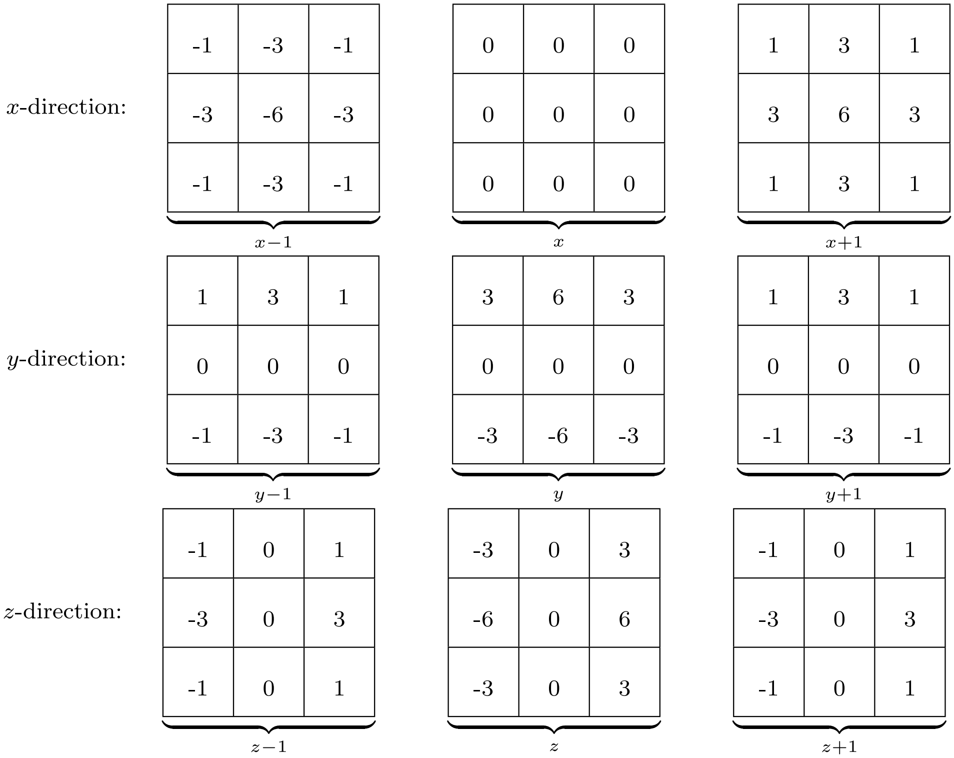
which uses different thresholds for different binary patterns, the proposed method is more adaptive to the image content.

The process is accomplished in two main paths applied within $3 \times 3 \times 3$ windows (blocks) centered at each voxel of the image. In the first path, a local threshold is selected to transform the gray-valued window into a binary map, so that the presence of an edge point is verified by means of binary patterns. The second path removes false edge points regarded in the first path mainly due to image intensity variation.

The main steps of the proposed method are illustrated in the diagram shown in Fig. 3. As it can be observed in the diagram, the technique is based on local and global thresholding.

Figure 4 depicts a 2D example of maps for each path of the method. The application of Boolean functions detects details of the images based only on the local values of a window. Therefore, relevant details are detected, including false edges. The second path detects regions for which the local value variation is higher than the noise level of the image. The application of the AND operator, using the two local responses, filters false edge responses and detects fine edge details.

The following sections describe the main aspects related to both stages of the proposed edge detection technique.



x-direction:

$x-1$			x			$x+1$		
-1	-3	-1	0	0	0	1	3	1
-3	-6	-3	0	0	0	3	6	3
-1	-3	-1	0	0	0	1	3	1

y-direction:

$y-1$			y			$y+1$		
1	3	1	3	6	3	1	3	1
0	0	0	0	0	0	0	0	0
-1	-3	-1	-3	-6	-3	-1	-3	-1

z-direction:

$z-1$			z			$z+1$		
-1	0	1	-3	0	3	-1	0	1
-3	0	3	-6	0	6	-3	0	3
-1	0	1	-3	0	3	-1	0	1

Fig. 2. $3 \times 3 \times 3$ Sobel masks.

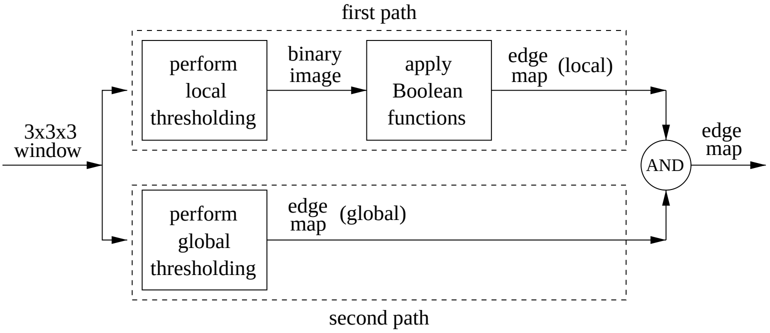


Fig. 3. Diagram of the proposed edge detection method. Both local and global thresholding are applied to $3 \times 3 \times 3$ windows centered at each voxel of the image.

3.1. *First path*

A $3 \times 3 \times 3$ window $W(x, y, z)$ centered at each (x, y, z) voxel of an $M \times N \times P$ image is used to calculate a local threshold $T_L(x, y, z)$. The threshold is based on Niblack's automatic local thresholding,[28] which is computed as the mean of local intensity values subtracted from a fraction of the standard deviation,

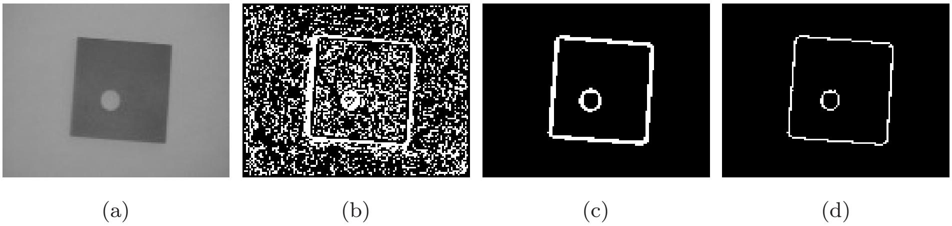


Fig. 4. Examples of responses expected for each path of the method. (a) Input image; (b) fine details of the first path; (c) regions with high standard deviation detected in the second path; (d) detected edges after ANDing first and second paths.

expressed by

$$T_L(x,y,z) = \mu(x,y,z) - \lambda\,\sigma(x,y,z), \tag{3}$$

where

$$\mu(x,y,z) = \frac{1}{3\times 3\times 3}\sum_{i=-1}^{1}\sum_{j=-1}^{1}\sum_{k=-1}^{1} W(x+i,y+j,z+k),$$

with $0 \le x \le M-1$, $0 \le y \le N-1$, $0 \le z \le P-1$, and λ is a real value that weights the standard deviation, σ.

However, both mean and standard deviation are sensitive to outliers, what may degrade the threshold estimation. A well-known alternative to the mean is the median. An alternative to standard deviation is the interquartile range, a measure of variability calculated as the difference between the 75th percentile and the 25th percentile. The interquartile range is a stable measure of data spread and is less affected by extreme values.[29] In the proposed method, local threshold is then computed as

$$T_L(x,y,z) = \text{median}(x,y,z) - \lambda\,\text{iqr}(x,y,z), \tag{4}$$

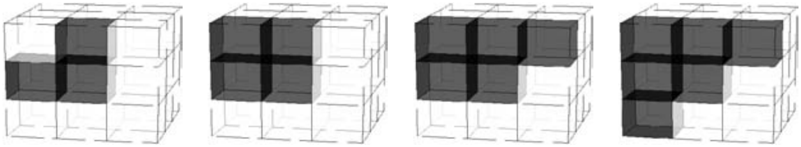
where iqr is the interquartile range measure of the window centered at (x,y,z). Special treatment, such as zero padding or repeated edge pixels, must be considered to handle image border.

The voxels $W(x,y,z)$ of the reference window W are thresholded as

$$W_L(x,y,z) = \begin{cases} 1, & \text{if } W(x,y,z) > T_L(x,y,z) \\ 0, & \text{otherwise,} \end{cases} \tag{5}$$

resulting a binary map of the window.

To find edge points, the binary map is filtered with a set of Boolean functions that represent edge patterns (masks). Figure 5 shows examples of patterns that are commonly found in object faces.



(a) (b) (c) (d)

Fig. 5. Examples of 3 × 3 × 3 patterns.

The patterns can be assigned to Boolean functions. For instance, the four patterns of Fig. 5 are stated as:

$$\mathrm{BF}_1 = V_1 \wedge \overline{V_2} \wedge V_3 \wedge \overline{V_4} \wedge \overline{V_5} \wedge V_6 \wedge V_7 \wedge V_8 \wedge V_9 \wedge \cdots \wedge V_{27}, \tag{6}$$

$$\mathrm{BF}_2 = \overline{V_1} \wedge \overline{V_2} \wedge V_3 \wedge \overline{V_4} \wedge \overline{V_5} \wedge V_6 \wedge V_7 \wedge V_8 \wedge V_9 \wedge \cdots \wedge V_{27}, \tag{7}$$

$$\mathrm{BF}_3 = \overline{V_1} \wedge \overline{V_2} \wedge \overline{V_3} \wedge \overline{V_4} \wedge \overline{V_5} \wedge V_6 \wedge V_7 \wedge V_8 \wedge V_9 \wedge \cdots \wedge V_{27}, \tag{8}$$

$$\mathrm{BF}_4 = \overline{V_1} \wedge \overline{V_2} \wedge \overline{V_3} \wedge \overline{V_4} \wedge \overline{V_5} \wedge V_6 \wedge \overline{V_7} \wedge V_8 \wedge V_9 \wedge \cdots \wedge V_{27}, \tag{9}$$

where $\wedge$ is the AND operator; the bar over an operand, $\overline{V_i}$, is the NOT operator; and each operand V_i is a binary voxel $W_L(x, y, z)$ in the $3 \times 3 \times 3$ window, numbered within the window as shown in Fig. 6.

If any pattern matches the window $W_L(x, y, z)$, the center element of the window is assigned as an edge point. Therefore, taking the already defined Boolean functions, a voxel is an edge point if the combination of the Boolean functions with the OR operation, $\vee$, is evaluated to 1 (that is, evaluated as true). The combination of the four Boolean functions is

$$\mathrm{BF} = \mathrm{BF}_1 \vee \mathrm{BF}_2 \vee \mathrm{BF}_3 \vee \mathrm{BF}_4. \tag{10}$$

The design of the Boolean functions is a crucial task for the performance of the method, since the combination of 0–1 patterns in three dimensions increases significantly in comparison to the two dimensional case. If the center pixel in 2D using a 3 × 3 Boolean binary function is set as object (assigned to 1), the number of possible 0–1 patterns in the 8-neighborhood is equal to 254 (disregarding the

	−1	0	1
−1	1	2	3
0	4	5	6
1	7	8	9

$z = -1$

	−1	0	1
−1	10	11	12
0	13	14	15
1	16	17	18

$z = 0$

	−1	0	1
−1	19	20	21
0	22	23	24
1	25	26	27

$z = 1$

Fig. 6. Mapping of window points.

two patterns with all 8 neighbors set as 0 or 1). On the other hand, the number of possible patterns can rise to 67108862 in 3D using a 3 × 3 × 3 window.

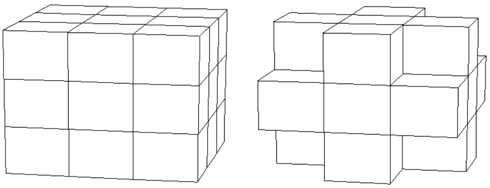
However, as pointed out by Vemis *et al.*[6] for the 2D case, the selection of the most common patterns met in practice usually leads to satisfactory results. Thus, 16 patterns out of 254 described in their work suffice for detecting edge points with a reduction factor of 16 with respect to the total number of patterns.

Similarly, patterns in 3D can be massively reduced. However, that is not such an easy work when compared to 2D. The combination of these 3D patterns makes the number of masks grow fast. It is worth noting that by neglecting some function constraints, larger sets can be expressed with fewer patterns. Instead of adopting the 26-neighborhood of a voxel (Fig. 7(a)) for the windows, if a 18-neighborhood (Fig. 7(b)) is considered, more patterns are covered with fewer masks. Enumerating all possibilities, there are 262142 patterns, approximately a reduction factor of 256 times. Therefore, the four patterns illustrated in Fig. 5 could be reduced to the one shown in Fig. 8.

The removed corners of the patterns can be thought as "don't cares" in the Boolean functions. Returning to the functions in Eqs. (6)–(9), the Boolean functions are obtained by removing the operands V_i and $\overline{V_i}$ with $i = 1, 3, 7, 9, 19, 21, 25, 27$. The four Boolean functions, as expected, reduce to the same Boolean function given as

$$\mathrm{BF}_1 = \overline{V_2} \wedge \overline{V_4} \wedge \overline{V_5} \wedge V_6 \wedge V_8 \wedge V_{10} \wedge \cdots \wedge V_{18} \wedge V_{20} \wedge V_{22} \wedge V_{23} \wedge V_{24} \wedge V_{26}. \tag{11}$$

The investigation of patterns in real images allows the identification of some simple edge patterns that, as will be seen, are important building blocks for other



(a) (b)

Fig. 7. Voxel neighborhood. (a) 26-neighborhood, (b) 18-neighborhood.

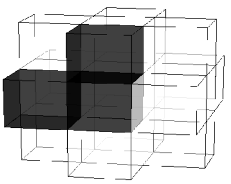


Fig. 8. Edge patterns encompassing the four patterns shown in Fig. 5.

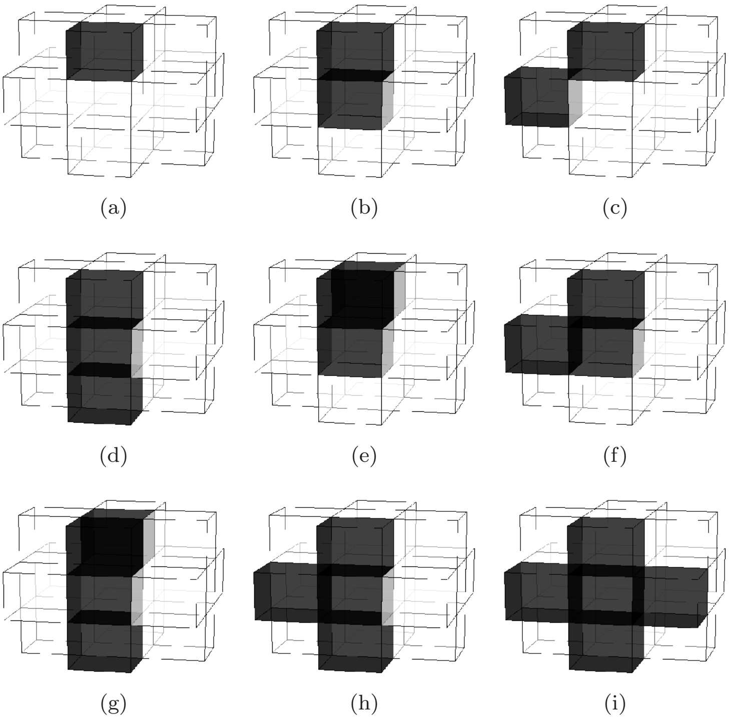


Fig. 9. Simple patterns.

common patterns. Such basic patterns are shown in Fig. 9. Rotations of these patterns produce masks to identify edges at different directions in the image.

As stated previously, only a small number of masks is necessary for specific cases of edge detection, for instance, objects containing exact horizontal and vertical faces. In particular, the patterns shown in Figs. 9(c), 9(f), 9(h) and 9(i) are well suited for such types of edge identification.

Nonetheless, for natural images, which often contain mostly irregular patterns, a small number of patterns usually does not suffice for a robust edge detection. Except for patterns shown in Figs. 9(c), 9(f), 9(h) and 9(i), these functions commonly do not produce proper edge detection results when individually employed. However, the most frequent patterns found in images are comprised of simpler patterns taken together for the construction of more specialized masks.

Examples of mask combinations are shown in Fig. 10. The compositions of the patterns shown in Figs. 9(h) and 9(i) with any other pattern (Figs. 10(a)–10(l)) are frequently found in natural images. Curved surfaces usually present patterns as the ones shown in Figs. 10(m)–10(p).

Enumeration of all masks results in 632 patterns, providing a reduction factor greater than 100000 times in comparison to the total of $3 \times 3 \times 3$ possible masks. The number of patterns can be easily held by using some fast indexing as a hash table.

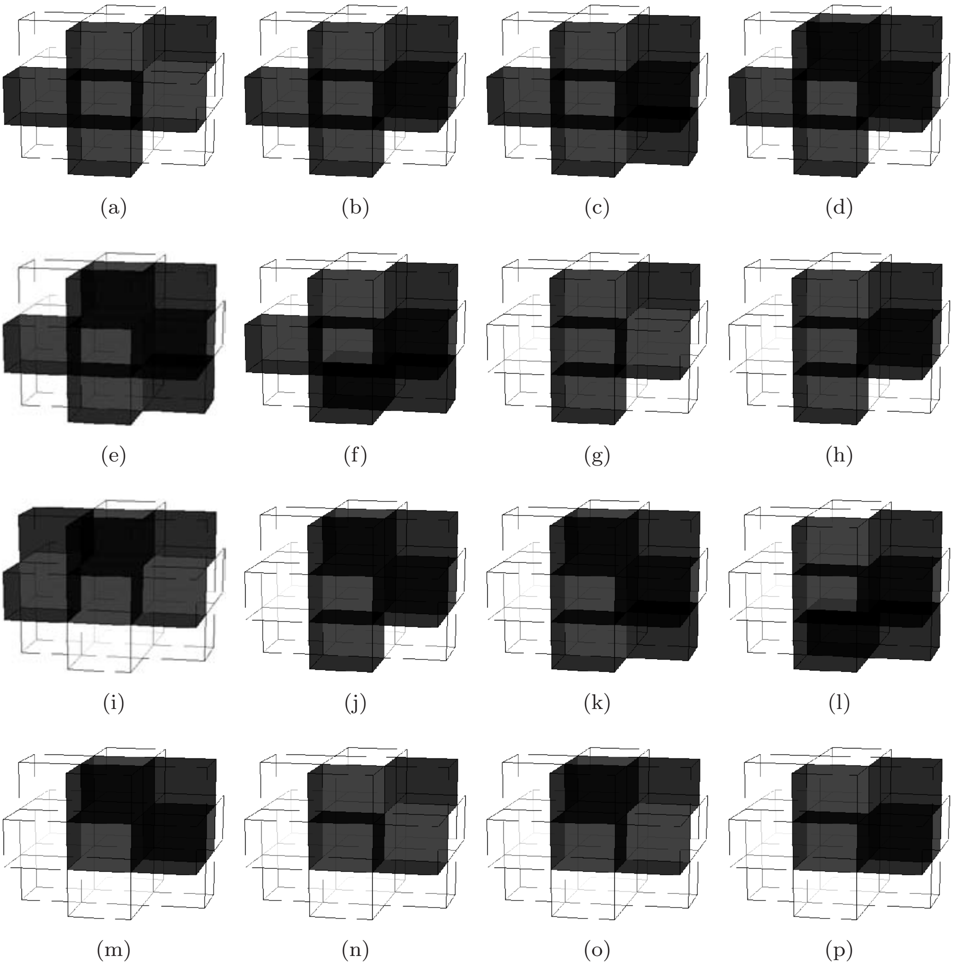


Fig. 10. Composed patterns.

3.2. *Second path*

The application of Boolean functions for each voxel of the volume results in an edge map. However, some false edges may be detected due to the presence of image intensity variation. A global thresholding T_G is used to remove these false edges. A new threshold, related to the intensity dispersion in the image, is estimated based on the image standard deviation weighted by a constant c as $T_G = c\,\sigma_I^2$, where

$$\sigma_I^2 = \frac{1}{M \times N \times P} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} \sum_{k=0}^{P-1} [f(i,j,k) - \mu_I]^2, f(i,j,k)$$

is the intensity value of the voxel (i,j,k), and μ_I is the mean value of all voxels in the image.

The interquartile range $\mathrm{iqr}(x, y, z)$ is locally calculated for each $3 \times 3 \times 3$ window $W(x, y, z)$, reaching its maximum value at an edge. This value is then compared to T_G and the central voxel is thresholded as

$$B_L(x,y,z) = \begin{cases} 1, & \text{if } \mathrm{iqr}\,(x,y,z) > T_G \\ 0, & \text{otherwise.} \end{cases} \tag{12}$$

Thus, if the value $B_L(x, y, z)$ is greater than the threshold T_G, $W_L(x, y, z)$ is kept as an edge, otherwise it is removed.

4. Experimental Results

Experiments and validations were performed using a set of 3D synthetic and real images. Synthetic images were employed to estimate how accurate the masks used in the proposed method fit to different types of edge patterns found in such images. Real images were used to visually and quantitatively evaluate the results obtained with the proposed method compared to state-of-the-art Canny, Monga–Deriche, Sobel and Zucker–Hummel edge detection methods.

Pratt's figure of merit[30] is widely employed to objectively rate the quality of edge detection,[31–34] defined as:

$$\mathrm{FOM} = \frac{1}{\max(N_I, N_D)} \sum_{i=1}^{N_D} \frac{1}{1 + \alpha d_i^2}, \tag{13}$$

where N_I and N_D are the numbers of ideal and detected edge pixels, respectively, α is an empirical constant (often 1/9) used to penalize displaced edges and d_i represents the distance between an edge point and the nearest ideal edge pixel. The value of figure of merit (FOM) is a number between 0 and 1, where 1 represents the best performance, that is, the detected edges coincide with the ideal edges. FOM was calculated for each method and used as a quantitative measurement to compare the obtained results.

Threshold selection is a crucial step for the performance of the proposed method. By examining the influence of thresholds on the outputs of several images used in the experiments, coefficient λ (first path of the algorithm) in Eq. (3) was set to 0.6 and global threshold T_G (second path of the algorithm) was empirically assigned to $0.6\sigma^2$.

Canny, Monga–Deriche, Sobel and Zucker–Hummel edge techniques are based on gradients and employ a thresholding to produce the binary edge image. Otsu's thresholding method[35] was used for Sobel and Zucker–Hummel methods as global thresholding, since it calculates an optimum threshold to separate classes. Canny and Monga–Deriche methods use two thresholds, a higher threshold and a lower one. The higher threshold identifies edges which are believed to be true edges. Otsu's threshold is then used for the higher threshold. The lower threshold was experimentally chosen to be 25% the value of Otsu's threshold.

To evaluate how well the masks fit to different kinds of edge patterns encountered in the images, the method was initially tested using only the first path on synthetic images. These images were generated along with their corresponding ground truth images, so that FOM value could also be measured. The synthetic images were composed of a black background with white regular and irregular objects, such as polyhedrons, rounded shapes and hand-free drawn objects.

The mean FOM response of each method applied to the synthetic images is summarized in Fig. 11. The ground truth for the FOM calculation was created by thresholding the original images with Otsu's thresholding method[35] and then applying a 27-neighborhood to detect edges. A voxel is assigned as edge element if it is an object voxel (set as 1) and at least one neighbor voxel belongs to background (set as 0).[36] The proposed method achieved FOM results comparable or superior to Canny, Monga–Deriche, Sobel and Zucker–Hummel methods. The measure is actually superior in mean terms for the tested images, since the identified edges are usually thinner than those obtained with the other two methods. This is an important advantage of the proposed method.

The use of synthetic images demonstrates that a suitable design of Boolean functions leads to better adaptivity to a variety of regular and irregular patterns. Three dimensional views of the detected edges for some of the tested images are shown in Fig. 12.

The next set of results is used for a qualitative and quantitative evaluation of the methods in real images. Three-dimensional images are commonly used for diagnostic purposes in medical applications, where samples are taken at different slices of organs and parts of the human body. Several DICOM brain volumes with 16-bit depth images were tested. These images are made available at site OsiriX[37] and the ground truth was computed as in the case of synthetic images previously

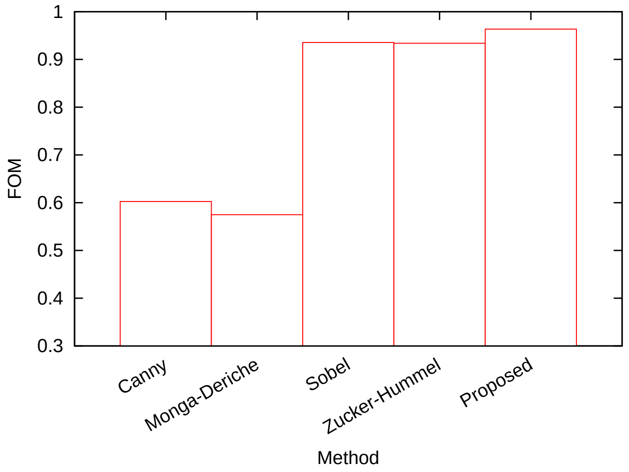
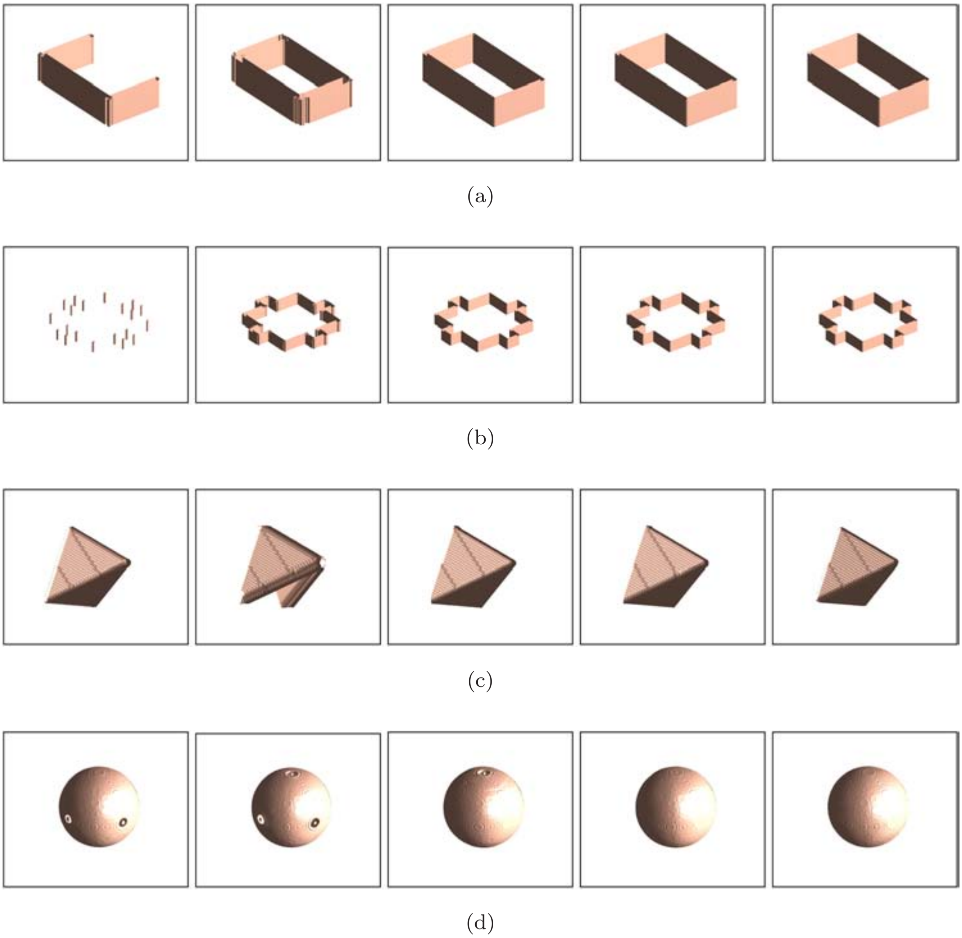


Fig. 11. Mean FOM for synthetic images obtained with tested methods.



(a)

(b)

(c)

(d)

Fig. 12. 3D views of edges detected by Canny, Monga–Deriche, Sobel, Zucker–Hummel and proposed method, respectively. (a) Parallelpiped, (b) Concave polyhedron, (c) tetrahedron and (d) ellipsoid.

described, that is, a voxel of the region (connected component) is considered as belonging to the boundary of a region if it has 27-neighborhood connectivity in the complement (background) of the region.

Information about the tested volume data sets and FOM measurements obtained by applying each edge detection method on real images are presented in Table 1. The proposed method, in mostly of the volume sets, outperforms Canny, Monga–Deriche, Sobel and Zucker–Hummel methods in terms of FOM, as highlighted in Table 1.

Two-dimensional visual results are illustrated in Fig. 13. Each group of figures shows six images: original and the five resulting edge maps obtained with Canny, Monga–Deriche, Sobel, Zucker–Hummel and proposed method, respectively.

Table 1. Mean FOM for real images obtained with tested methods.

			FOM				
Volume data sets	Size (voxels)	Depth (bits)	Canny	Monga–Deriche	Sobel	Zucker–Hummel	Proposed
Brain	256 × 256 × 30	16	0.8136	0.7342	**0.8907**	0.8645	0.8378
Disk	256 × 256 × 9	8	0.8113	0.1480	0.9178	0.9129	**0.9670**
Leg	400 × 400 × 30	24	0.8119	0.8078	0.8290	0.8287	**0.8776**
Liver	256 × 256 × 30	24	0.9012	0.8139	0.8916	0.8958	**0.9206**
Knee1	256 × 256 × 14	24	0.8732	0.6511	0.8881	**0.8947**	0.8918
Heart	256 × 256 × 30	24	0.8516	0.8473	**0.8879**	0.8730	0.8477
Knee2	256 × 256 × 24	24	0.9077	0.7891	0.7673	0.9071	**0.9525**
Knee3	256 × 256 × 9	24	0.8430	0.6403	**0.9093**	0.9017	0.8593

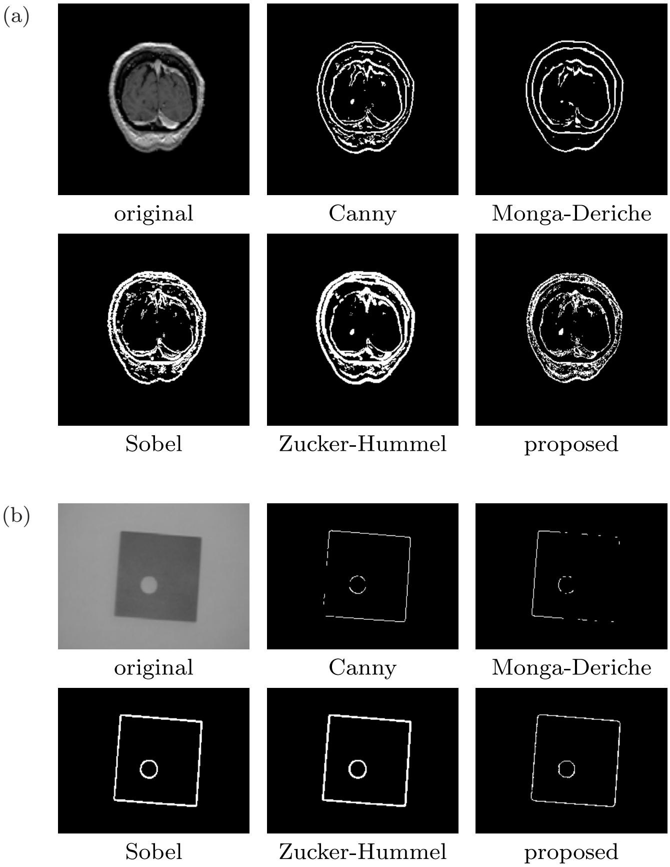


Fig. 13. Original volume frames and corresponding edge maps obtained with each edge detection method. (a) Brain, (b) disk, (c) leg, (d) liver, (e) knee1, (f) heart, (g) knee2 and (h) knee3.

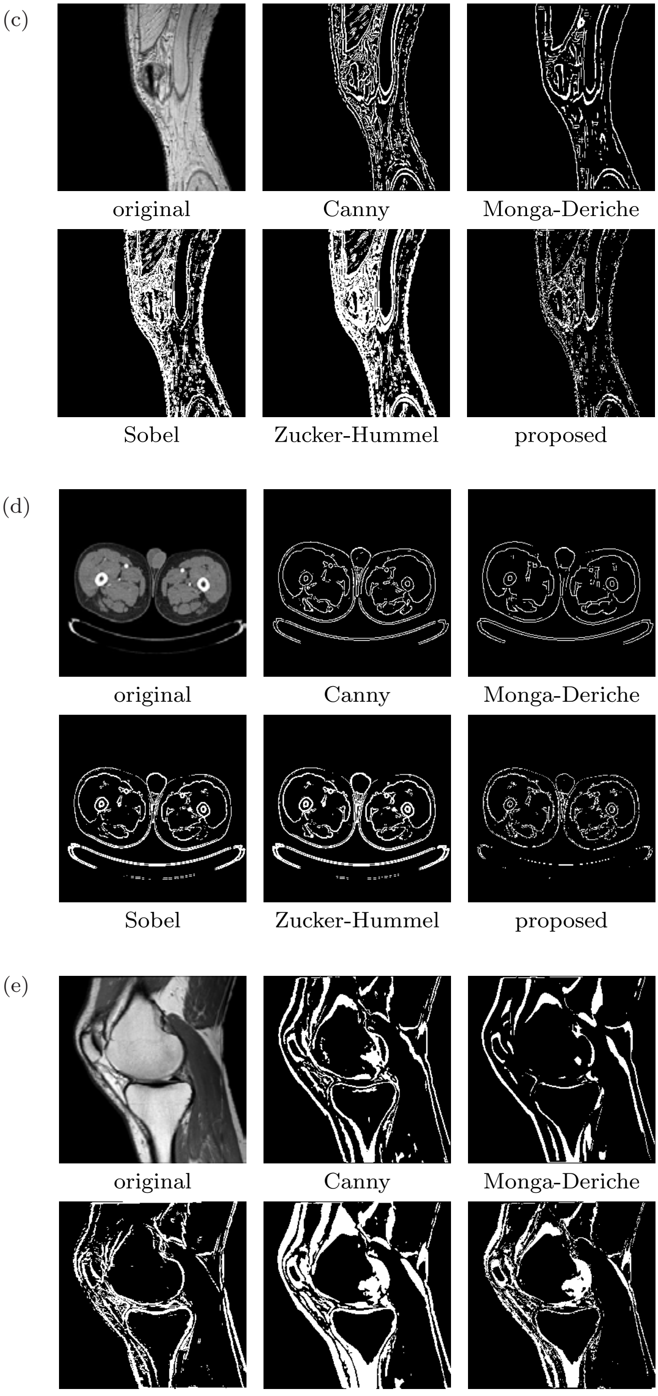


Fig. 13. (*Continued*)

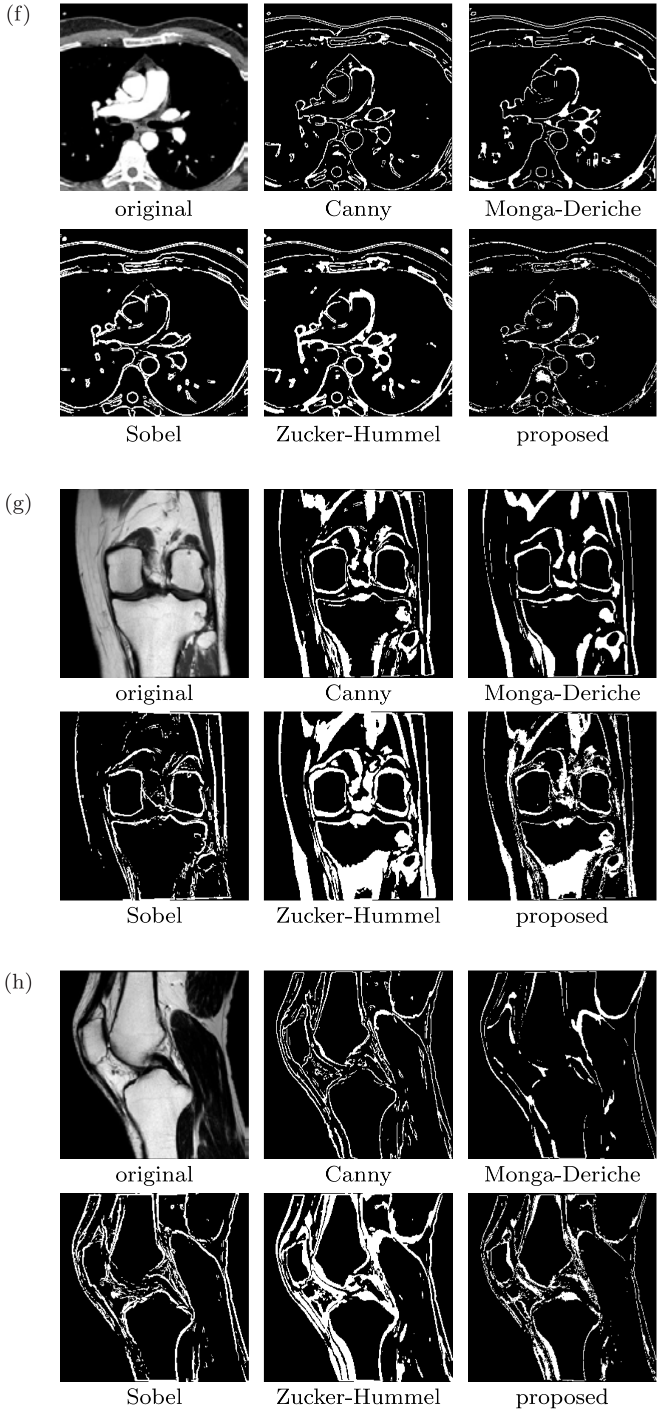
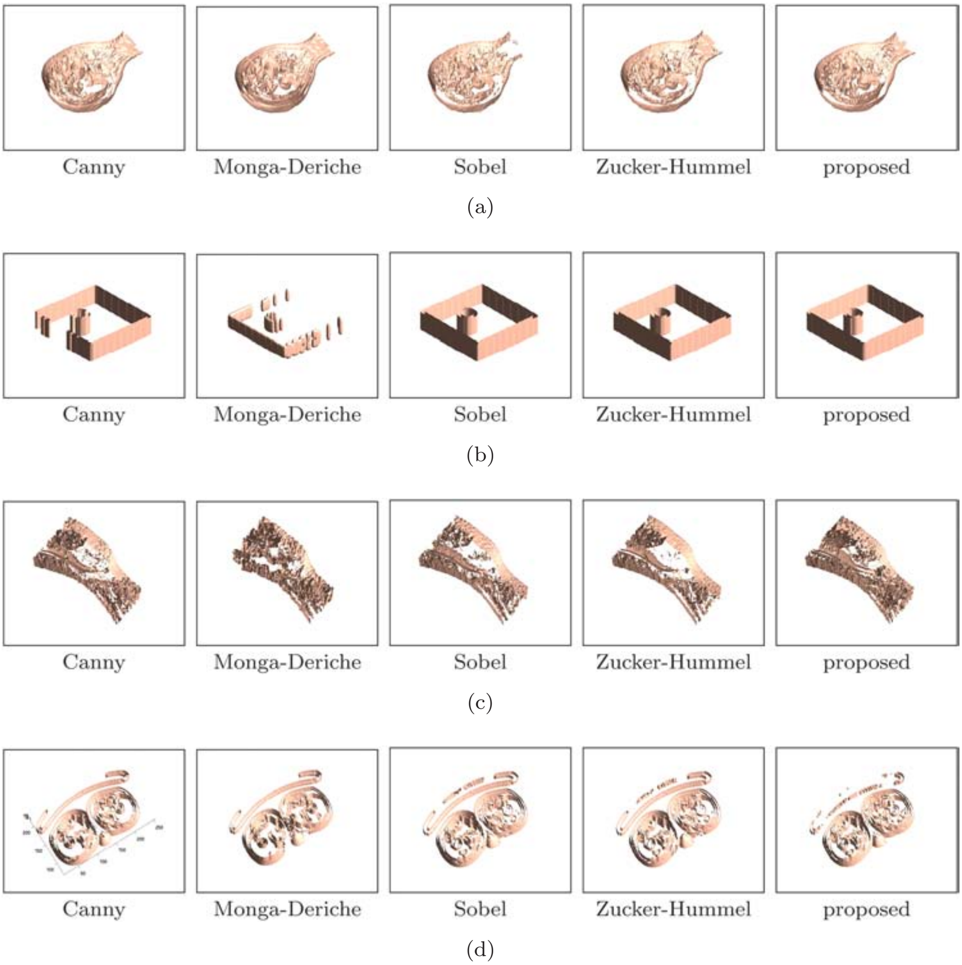


Fig. 13. (*Continued*)

From the images, it is possible to note that our method identifies detailed edge features of the medical volumes with results comparable or superior to Canny, Monga–Deriche, Sobel and Zucker–Hummel. Furthermore, our method produces edges with thinner lines, which is more suitable for a number of applications.

Another effect that should be noticed is the presence of large connected regions as the ones in Figs. 13(e) and 13(g), corresponding to surfaces (edges) identified from adjacent slices.

Three-dimensional views of the edges are shown in Fig. 14, where the images in each row correspond to the edge maps obtained with Canny, Monga–Deriche, Sobel, Zucker–Hummel and proposed method, respectively. As can be seen in Fig. 14(g),



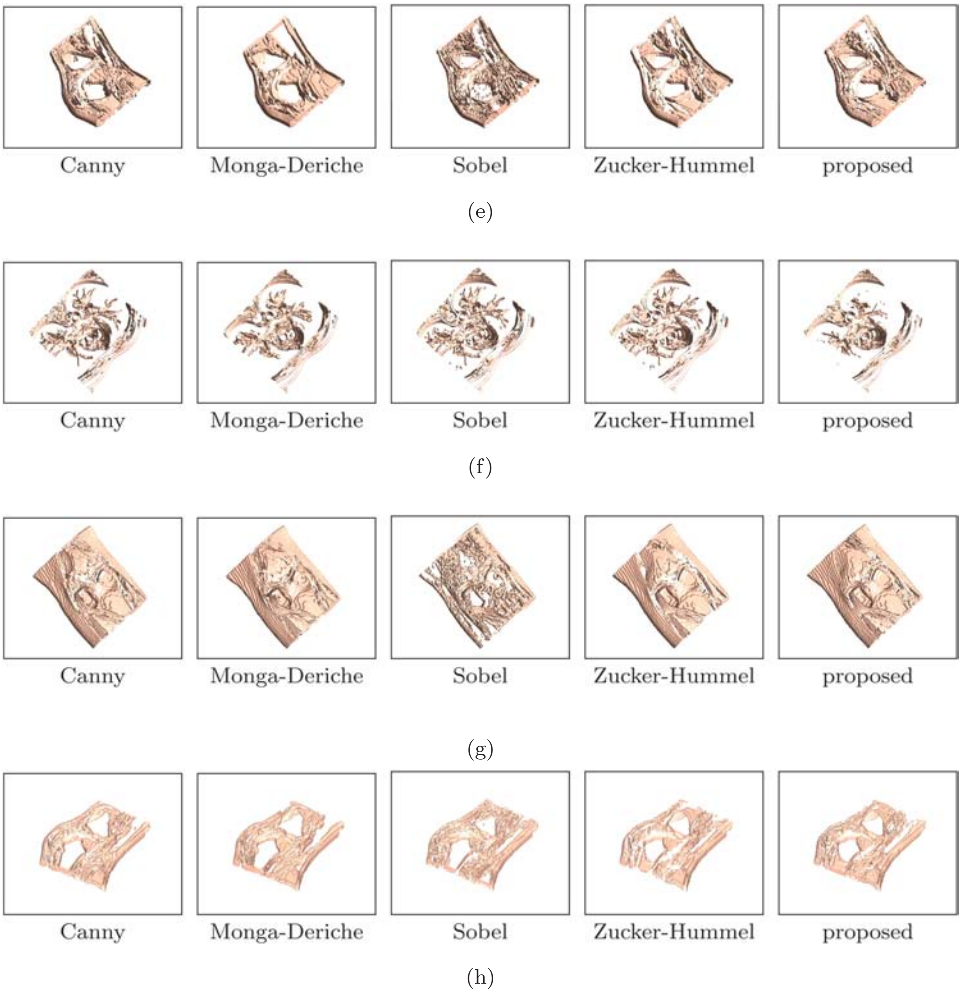
(a)

(b)

(c)

(d)

Fig. 14. View of the 3D edge detection results for data sets. (a) Brain, (b) disk, (c) leg, (d) liver, (e) knee1, (f) heart, (g) knee2 and (h) knee3.



(e)

(f)

(g)

(h)

Fig. 14. (*Continued*)

these edges are well identified by Canny, Monga–Deriche, Zucker–Hummel and the proposed method, while Sobel presents a poor response to such regions.

The proposed method requires an adequate design of the masks for the Boolean functions to cover as many as possible edge patterns in the image. This design flexibility offers the possibility of including additional Boolean functions to detect specific features of the images.

5. Conclusions and Future Work

Analysis of 3D images is an important task for several knowledge domains, such as medicine, biology, industry and geology. Edges are considered as relevant features for image understanding.

The method proposed in this work detects edges in 3D data sets using two main stages. In the first step, images are locally thresholded and cross-correlated with edge patterns through Boolean functions. The second step removes false edges based on image intensity variation. Both stages are combined to generate the final edge map.

The proposed method was tested on several synthetic and real data sets and compared to Canny, Monga–Deriche, Sobel and Zucker–Hummel edge detectors. The new method allows to include Boolean functions according to a specific need, providing flexibility for edge detection.

As directions for future work, a cluster architecture could be considered for the implementation of the edge detection methods to improve their computational speed. Representing images through their edges has the advantage of reducing the amount of data required to store and transmit relevant image information. In large data sets, the use of parallel programming can significantly speed up the algorithms. On the other hand, the search for more accurate edge detection methods is crucial, since they can be employed as an important step in a variety of applications, such as face recognition, object tracking, image segmentation, among others.

Acknowledgments

The authors are thankful to FAPESP, CNPq and CAPES for the financial support.

References

1. R. da Silva, R. Minetto, W. Schwartz and H. Pedrini, “Adaptive edge-preserving image denoising using wavelet transforms,” *Pattern Analysis & Applications* **16**(4), 567–580 (2013).
2. F. Cao, P. Musé and F. Sur, “Extracting meaningful curves from images,” *Journal of Mathematical Imaging and Vision* **22**(2–3), 159–181 (2005).
3. A. Desolneux, L. Moisan and J.-M. Morel, “Edge detection by Helmholtz principle,” *Journal of Mathematical Imaging and Vision* **14**(3), 271–284 (2001).
4. P. S. C. Bähnisch and U. Köthe, “Fast and accurate 3D edge detection for surface reconstruction,” *Lecture Notes in Computer Science* **5748**, 111–120 (2009).
5. E. Meinhardt, E. Zacur, A. Frangi and V. Caselles, “3D Edge detection by selection of level surface patches,” *Journal of Mathematical Imaging and Vision* **34**(1), 1–16 (2009).
6. M. Vemis, G. Economou, S. Fotopoulos and A. Khodyrev, “The use of Boolean functions and local operations for edge detection in images,” *Signal Processing* **45**(2), 161–172 (1995).
7. M. Basu, “Gaussian-based edge-detection methods: A survey,” *IEEE Transactions on Systems, Man, and Cybernetics* **32**(3), 252–260 (2002).
8. L. Davis, “A survey of edge detection techniques,” *Computer Graphics and Image Processing* **4**(3), 248–270 (1975).
9. E. Nadernejad, S. Sharifzadeh and H. Hassanpour, “Edge detection techniques: Evaluations and comparisons,” *Applied Mathematical Sciences* **2**(31), 1507–1520 (2008).
10. D. Ziou and S. Tabbone, “Edge detection techniques: An overview,” *International Journal of Pattern Recognition and Image Analysis* **8**(4), 537–559 (1998).

11. L. G. Roberts, “Machine perception of three-dimensional solids,” *Optical and Electro-Optical Information Processing*, J. T. Tippett, D. A. Berkowitz, L. C. Clapp, C. J. Koester and A. Vandenburgh Jr., eds., MIT Press, Cambridge, MA, USA, pp. 159–197, 1965.
12. J. M. S. Prewitt, “Object enhancement and extraction,” *Picture Processing and Psychopictorics*, B. S. Lipkin and A. Rosenfeld, eds., Academic Press, New York, NY, USA, pp. 75–149, 1970.
13. I. Sobel, *Camera Models and Machine Perception*, Ph.D. thesis, Stanford University, Stanford, CA, USA, May 1970.
14. D. Marr and E. Hildreth, “Theory of edge detection,” in *Proc. Royal Society of London, Series B, Biological Sciences*, Vol. 207 (1980), pp. 187–217.
15. J. Canny, “A computational approach to edge detection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence* **8**(6), 679–698 (1986).
16. R. Deriche, “Using Canny’s criteria to derive an optimal edge detector recursively implemented,” *International Journal on Computer Vision* **1**, 167–187 (1987).
17. O. Monga and R. Deriche, “3D edge detection using recursive filtering: Application to scanner images,” *CVGIP — Image Understanding* **53**(1), 76–87 (1991).
18. H. Liu, “Two- and three-dimensional boundary detection,” *Computer Graphics and Image Processing* **6**(2), 123–134 (1977).
19. D. Morgenthaler and A. Rosenfeld, “Multidimensional edge detection by hypersurface fitting,” *IEEE Transactions on Pattern Analysis and Machine Intelligence* **3**(4), 482–486 (1981).
20. L. Luo, C. Hamitouche, J. Dillenseger and J. Coatrieux, “A moment-based three-dimensional edge operator,” *IEEE Transactions on Biomedical Engineering* **40**(7), 693–703 (1993).
21. P. Bhattacharya and D. Wild, “A new edge detector for gray volumetric data,” *Computers in Biology and Medicine* **26**(4), 315–328 (1996).
22. O. Monga and S. Benayoun, “Using partial derivatives of 3D images to extract typical surface features,” *Computer Vision and Image Understanding* **61**(2), 171–189 (1995).
23. S. Zhan and R. Mehrotra, “A zero-crossing-based optimal three-dimensional edge detector,” *CVGIP — Image Understanding* **59**, 242–253 (1994).
24. Y. Zhang, “Quantitative study of 3D gradient operators,” *Image and Vision Computing* **11**(10), 611–622 (1993).
25. S. Zucker and R. Hummel, “An optimal three-dimensional edge operator,” Technical report, McGill University, Toronto, Ontario, Canada, April 1979.
26. S. Zucker and R. Hummel, “A three-dimensional edge operator,” *IEEE Transactions on Pattern Analysis and Machine Intelligence* **3**(3), 324–331 (1981).
27. M. Hueckel, “An operator which locates edges in digitized pictures,” *Journal of the ACM* **18**(1), 113–125 (1971).
28. W. Niblack, *An Introduction to Image Processing* (Prentice Hall, Inc., Englewood Cliffs, NJ, USA, 1986).
29. W. Gilchrist, *Statistical Modelling with Quantile Functions* (Chapman & Hall/CRC, Boca Raton, FL, USA, 2000).
30. W. Pratt, *Digital Image Processing* (Wiley-Interscience, New York, NY, USA, 1978).
31. S. Coleman, B. Scotney and S. Suganthan, “Multi-scale edge detection on range and intensity images,” *Pattern Recognition* **44**(4), 821–838 (2011).
32. A. Kaur and C. Singh, “A hybrid edge detector using fuzzy logic and mathematical morphology,” *International Journal of Image and Graphics* **10**(2), 251–272 (2010).

33. R. LeAnder, M. Chowdary, S. Mokkapati and S. Umbaugh, "Comparison of two algorithms in the automatic segmentation of blood vessels in fundus images," in *Proc. SPIE*, Vol. 6915, April 2008.
34. R. Moreno, M. Garcia, D. Puig and C. Julia, "Robust color edge detection through tensor voting," in *16th IEEE International Conference on Image Processing* Cairo, Egypt, November 2009, pp. 2153–2156.
35. N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Transactions on Systems*, *Man and Cybernetics* **9**(1), 62–66 (1979).
36. R. C. Jain, R. Kasturi and B. G. Schunck, *Machine Vision* (McGraw-Hill, 1995).
37. OsiriX, DICOM Sample Image Sets, 2010. http://www.osirix-viewer.com/datasets/.

Ricardo Dutra da Silva received his Ph.D. degree in Computer Science from University of Campinas and his M.Sc. and B.Sc. degrees in Computer Science from Federal University of Paraná. His research interests include image processing and analysis, computer vision, computational geometry and computer graphics.

Rosane Minghim received her Ph.D. degree in Computer Studies from University of East Anglia, UK, and her M.Sc. degree in Electrical Engineering from University of Campinas, Brazil. She is an associate professor at University of São Paulo, São Carlos, Brazil. She is interested in all aspects of visualization, information visualization and visual analytics.

Helio Pedrini received his Ph.D. degree in Electrical and Computer Engineering from Rensselaer Polytechnic Institute, Troy, NY, USA. He received his M.Sc. degree in Electrical Engineering and his B.Sc. in Computer Science, both from University of Campinas, Brazil. He is currently a professor in the Institute of Computing at the University of Campinas, Brazil. His research interests include image processing, computer vision, pattern recognition and computer graphics.