CHAPTER 5

# CEPHALOMETRIC EDGE DETECTION

## 5.1. Introduction

Edge detection is one of the fundamental steps in image analysis and is useful in finding the key features in images. An edge is defined by a discontinuity in gray level values. Factors such as image contrast, sharpness, complexity of image structures, illumination conditions, the noise level in the images affects the results of an edge detector. To improve the reliability of subsequent processing steps it is needed that the edge detection results must be efficient and reliable. Traditional edge detectors like Sobel and Prewitt are sensitive to noise and blur in the images. To overcome these drawbacks advance edge detectors are proposed in the literature like Canny and Susan. However, even these edge detectors are not highly affective for extremely complex medical images like cephalometric X-ray images that suffer from very low contrast in certain areas, nonuniform illumination and overlapped complicated soft tissue and bony structures. Modeling information uncertainty and subjectivity are difficult in these algorithms. Fuzzy logic allows us to model uncertainty and subject concepts in a better form than crisp models. Thus in Chapter we analyze the existing fuzzy logic based edge detection methods and proposed a simple fuzzy based edge detector for cephalometric images. The results are compared with Canny, Susan and other promising fuzzy techniques.

In case of cephalometric analysis algorithms, Tong et al. [10] used Prewitt's operator for edge enhancement. Grau et al. [24] applied to log for edge detection. Eli-Feghi et al. [26] applied Sobel's operator to enhance the edges. Yue et al. [17] extracted edges using the Canny edge detector. Kafieh et al. [27] detected edges using Susan edge detection and morphological opening. Favaedi et al. [33] used Canny edge detection to isolate the edges. From the review of the existing literature, it is evident that most algorithm except [27] used traditional edge detection techniques based on first and second order derivates. The authors in [27] apply Susan edge detection technique to get better noise resistance and faster execution than Canny.

## 5.2. Pixel Level Edge Detection

The majority of traditional edge detectors are linear operators that are derivatives of some sort of smoothing filter. Sobel and Prewitt detectors are one of the initial methods that used local gradient operators to detect the edges. Their main limitation is that they perform poorly when the edges are blurred and noisy. Since then, more sophisticated operators were developed to provide better edge detection results. Canny is the most widely used operator in this category. This method uses Gaussian smoothing to reduce the effect of noise before detecting the edges [35]. However, the presence of noise can still degrade its performance, and it may require many more runs using different combinations of parameters (the upper and lower threshold values and sigma value) although a very good tradeoff can be found after enough runs. Furthermore, at times this edge detector tends to connect lines into closed contours and may give rise to unnecessary edges thus cluttering the main shape details.

### 5.2.1. Fuzzy edge detection

Image processing algorithms are based on crisp logic, thus information uncertainty and subjectivity are difficult to model in the algorithms. Fuzzy logic allows to model uncertainty and subject concepts in a better form than crisp models. More recently, fuzzy techniques have been proposed to introduce new approaches to edge detection that characterizes edge detection as a fuzzy reasoning problem [64-69]. Mathematical morphology provides an alternative approach to image processing based on the shape concept stemmed from set theory. The basic morphological operations namely erosion, dilation, opening, closing, etc. are widely used for detecting, modifying, manipulating the edge features present in the image based on their shapes. In this work we use, mathematical morphology for edge thinning [35].

### 5.2.2. Proposed algorithm

The proposed algorithm has two parts, edge detection using fuzzy technique and edge thinning based on binary mathematical morphology. The algorithm is given as follows:

### 5.2.2.1. Edge detection using fuzzy logic

Fuzzy logic represents a good mathematical framework to detect edges. It can deal with situations in, which making a sharp distinction between the boundaries is difficult as in case of image edge detection. The technique has four steps.

**Step I:** For each pixel $P_{ij}$ in the image horizontal $G_h$, vertical $G_v$, and diagonal $G_d$ edge strength are found using spatial window of the size $3 \times 3$ (Fig. 5.1).

$$G_h = |H_1 - H_2| \tag{5.1}$$

$$G_v = |V_1 - V_2| \tag{5.2}$$

$$G_d = |D_1 - D_2| \tag{5.3}$$

The selection of window size is influenced by other similar approaches in the literature [65-67] which use $3 \times 3$ window $W$.

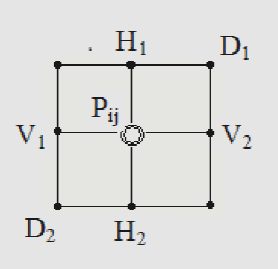



Figure 5.1: The centre circle corresponds to the pixel $p_{ij}$ whose fuzzy value has to be calculated and the eight surrounding black dots are its eight neighbors $(N_{xy})$ that are used in the calculation of variance.

**Step II:** The degree of edginess $\hat{\mu}_F$ for the pixel is then computed from the fuzzy membership value in horizontal $\mu_h$, vertical $\mu_v$, and diagonal $\mu_d$ direction. The process is explained below

$$\mu_{\mathrm{h}} = \frac{\mathrm{G_h}}{\max(\mathrm{w})} \tag{5.4}$$

$$\mu_{\mathrm{v}} = \frac{G_v}{max(w)} \tag{5.5}$$

$$\mu_{\mathrm{d}} = \frac{\mathrm{G_d}}{\max(\mathrm{w})} \tag{5.6}$$

$$If\ max(w) - min(w) > Th$$

$$\hat{\mu}_F = max(\mu_h, \mu_v, \mu_d) \tag{5.7}$$

$$else$$

$$\hat{\mu}_F = 0$$

$$end$$

This step gives a fuzzified image, which is saved in buffer $F\_Buf$.

**Step III:** Apply complement operation on the fuzzified image stored in $F\_Buf$, which will give the negated image stored in $NF\_Buf$. If $A$ is the fuzzy set, its complement ¬A can be found as follows

$$\mu\neg A(x) = 1 - \mu A(x) \tag{5.8}$$

**Step IV:** After this, the T-norm operation, which is a fuzzy conjunction operation, is applied on the two fuzzy images stored in $F\_Buf$ and it's negative $NF\_Buf$. There are many choices for fuzzy conjunction. In case of the proposed algorithm, the T-norm based on $min$ operation is considered. In the literature of fuzzy sets, $min$ operation plays an important role. This fuzzy T-norm performs exactly same as the intersection in crisp sets. The fuzzy set possesses certain properties that make it special for this application. The most important properties are

(i) Idempotency ($T(x, x) = x$) for all $x \in [0, 1]$

(ii) This T-norm produces the largest fuzzy set from those produced by all possible fuzzy intersections for a given fuzzy set. The fuzzy intersection of two fuzzy sets $A$ and $B$ on the universe of discourse $X$ is defined by

$$\mu A \cap B(x) = min[\mu A(x), \mu B(x)] = \mu A(x) \cap \mu B(x), \quad x \epsilon X \tag{5.9}$$

This gives the final fuzzified image those fuzzy values give the edge information.

**Step V:** Depending on the fuzzy membership value of each pixel decision is made if it is an edge pixel or a background pixel, by using the edge detail parameter $Th$. A value of zero is assigned to edge pixels and a value of 255 to all the background pixels. By adjusting the value of $Th$, different levels of edge details can be obtained.

#### 5.2.2.2. Morphology based image thinning

Binary mathematical morphological operation is applied to obtain thin edges. Fuzzy morphology is not efficient in edge thinning that has a strong dependency on the presence of outliers as in the case of binary mathematical morphology. Edge thinning is applied on the edge image obtained by applying the fuzzy edge detection technique. The thinning operation is based on hit-or-miss transform (HMT). The thinning of the image $I$ is defined by Eq. (5.10).

$$I \otimes J = I - (I \Theta J) \tag{5.10}$$

Where $I \otimes J$ corresponds to edge thinned image, $I$ is the input image. The composite structuring element and $(I \Theta J)$ is the result obtained after applying HMT to the

image $I$. The composite structuring element consists of many component structuring elements $(J_1, J_2, J_3, \ldots J_n)$ where $J_2$ is the rotated version of $J_1$ and $J_3$ is rotated version of $J_2$ and the same is true for and all the component structuring elements. As is clear from Fig. 5.2 (a), if we consider $J_1$ as a component structuring element and rotate its origin by $90^o$ then we get $J_2$ shown in Fig. 5.2 (b). All the structuring elements $(J_1, J_2, J_3, \ldots J_n)$ in the composite structuring element must satisfy the following equation

$$J_1 \cap J_2 \cap J_3 \ldots \cap J_n = \phi \tag{5.11}$$

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | | 0 | 0 | 0 |
| 0 | 1 | 1 | | 1 | 1 | 0 |
| 0 | 1 | 0 | | 0 | 1 | 0 |
| | (a) | | | | (b) | |

Fig. 5.2: (a) Structuring element and (b) $90^o$ rotated version of (a).

Thinning by a sequence of structuring elements $(J_1, J_2, J_3, \ldots J_n)$ is given below

i. Translate the origin of the structuring element $J_i$ to each pixel in the edge image $(ed\_image)$.

ii. Apply erosion and dilation operations on the edge image and obtain the eroded edge image $(eout)$ and dilated edge image $(dout)$.

iii. Perform HMT using the results from step (ii) using the following equation

$$hout = eout \& \sim dout \tag{5.12}$$

The partial edge thinning is obtained by using $hout$ from step (iii) and subtracting it from the edge image$(ed_image)$. The initial $ed\_image$ is the edge image obtained by applying fuzzy edge detection technique.

$$ed\_image = |\, ed\_image - hout \,| \tag{5.13}$$

iv. Steps i, ii, iii, and iv are repeated with each component structuring element of $J$.

v. The final edge thinned image is obtained by iterating until convergence is achieved.

### 5.2.3. Results and discussions

To highlight the performance of the proposed algorithm, we adopted 20 gray scale test images (synthetic and natural images) of various sizes, contrasts, and with varying edge sharpness and 20 cephalometric X-ray image (ten digital and ten scanned). The results yielded by the proposed algorithm are compared with other fuzzy based edge

detectors and with Canny and Susan edge detectors that are widely used standard edge detection techniques. The visual results corresponding to test image Fig. 5.3 (a) are shown in Fig. 5.4.

The visual results for a cephalometric test image Fig. 5.3 (b) are shown in Fig. 5.5. The results are obtained by using only a single iteration for edge thinning. The visual performance of the proposed algorithm is clearly perceptible. From the visual analysis of Fig. 5.4 and Fig. 5.5 it becomes clear that the proposed algorithm gives results that look highly natural, suffer from very few false and missing edges. In all the test images, the original shape is retained with a good balance of detail and the edge localization. In case of test image, 'peppers' (Fig. 5.4) Tizhoosh's method I give very clean and smooth edges but tend to miss certain edges, and the edges are quite thick. His results for method II and method III give more complete edges but suffer from some noise owing to reflection effects in the image. Jinbo's and Liang's results give thinner edges, but the derived edges are noisy. Results from Russo's method are similar to Tizhoosh's method I. The results given by Canny, Susan and the proposed algorithm are better than the above methods. The results of the proposed algorithm are better than the Canny and Susan edge detector. Canny tends to connect some lines into closed contours, cluttering the main shape, the proposed algorithm does not clutter the principal shape though it losses few insignificant edges. Susan edge detector gives results closer to the proposed algorithm but misses more edges. If the brightness threshold is reduced for Susan the visibility of spurious edge's increases.

Figure 5.3: Test image (a) Peppers and (b) Cephalogram.


Figure 5.4: This figure gives the edge results corresponding to test image pepper (a) Tizhoosh's method I (b) Tizhoosh's method II (c) Tizhoosh's method III (d) Jinbo's algorithm (e) Liang's fuzzy edge detector (f) Russo's method (g) Canny edge detector (h) Susan edge detector and (i) Proposed algorithm.

The test result for a cephalogram test image Fig. 5.3 (b) is given in Fig. 5.5. The image is highly complex, and the contrast between the background and soft tissue is very low (nose region). The image suffers from nonuniform illumination effects in the chin region. The internal structures are complex due to overlap of structures. For cephalograms, most methods except Tizhoosh's method III and the proposed algorithm misses soft tissue edges. Tizhoosh's method I, Jinbo's and Liang's method gives very noisy results with many false edges, and many significant edges are lost. Tizhoosh's method II, Russo's and Susan's results are not very noisy but miss many

important edges. Canny gives many unwanted edges with closed loops in regions of subtle gray scale changes and misses low contrast edges. The resulting image gives a very unnatural look.

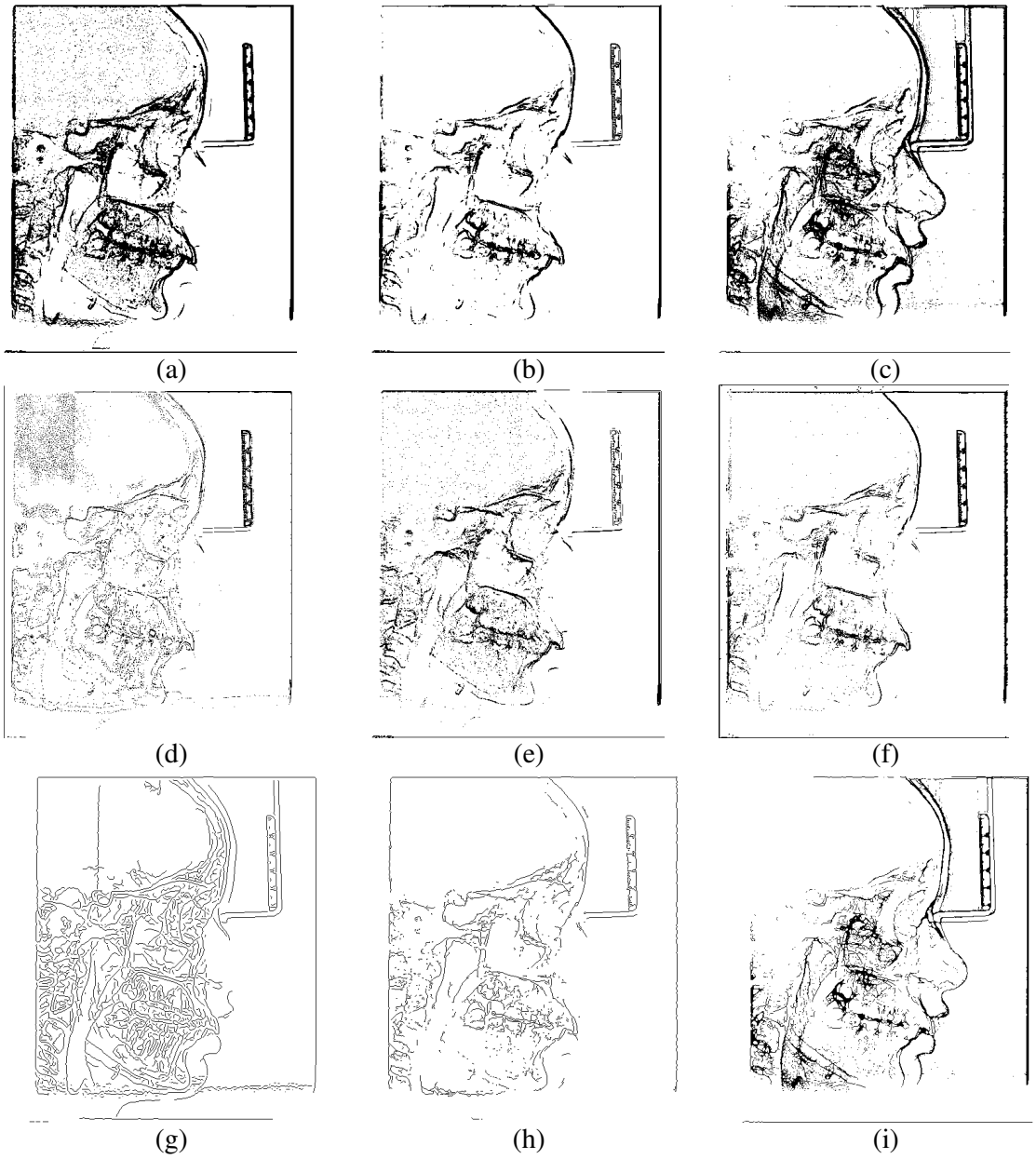



Figure 5.5: This figure gives the edge results corresponding to cephalogram test image (a) Tizhoosh's method I (b) Tizhoosh's method II (c) Tizhoosh's method III (d) Jinbo's algorithm (e) Liang's fuzzy edge detector (f) Russo's method (g) Canny edge detector (h) Susan edge detector and (i) Proposed algorithm.

The proposed algorithm gives best results compared to the other methods with most significant edges present with fewer noisy edges. The obtained images represent the

structures present in the image in a better way though not all edges are single pixel thin as in Canny and Susan. The performance of our algorithm is better than Canny in certain cases as in case of test image shapes (Fig. 5.6). The result of the Canny edge detector gives some distortions, specifically near the corners whereas Susan gives better results than Canny, and the proposed algorithm gives the best results the algorithm presents clean edges.

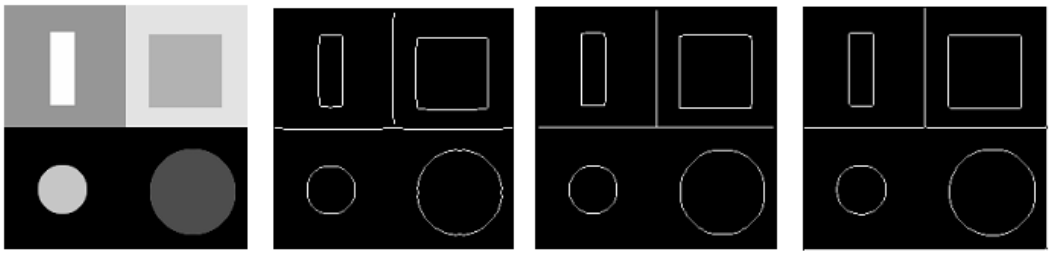



Figure 5.6: (a) Test image shapes. Results obtained by (b) Canny edge detector (c) Susan edge detector and (d) Proposed algorithm.

The Canny edge detector has four main passes Gaussian blurring using $(5 \times 5)$ window, finding gradient strength and direction using Sobel's mask $(3 \times 3)$, tracing along the edges and nonmaximum suppression whereas the proposed edge detector has only two passes one for fuzzy edge detection using $(3 \times 3)$ window and the other for edge thinning using mathematical morphology again using $(3 \times 3)$ window, thus the computational burden of Canny is higher than the proposed algorithm. The time efficiency of all the algorithms is found by executing corresponding c programs on PC with 2.8 GHz CPU and 1GB RAM. The computation time of each algorithm is given in Table 5.1. The computation time of the proposed algorithm is better than most fuzzy edge detectors and Canny edge detector except Tizhoosh's method I and II, as is clear from the Table 5.1.

**Table 5.1 Computation time of the algorithms in milliseconds (msec).**

| **Algorithms** → Test Images (pixel × pixel) | Tizhoosh I | Tizhoosh II | Tizhoosh III | Jinbo | Liang | Russo | Canny | Susan | Proposed algorithm |
|---|---|---|---|---|---|---|---|---|---|
| T1 (309 ×314) | 078 | 063 | 141 | 156 | 093 | 141 | 094 | 069 | 063 |
| T2 (475 ×357) | 125 | 093 | 156 | 218 | 125 | 203 | 141 | 105 | 125 |
| T3 (500 ×362) | 063 | 078 | 078 | 156 | 078 | 156 | 062 | 046 | 047 |
| T4 (309 ×314) | 125 | 141 | 188 | 235 | 125 | 219 | 156 | 115 | 141 |
| T5 (472×467) | 110 | 100 | 219 | 250 | 141 | 235 | 187 | 139 | 152 |
| T6 (200×200) | 016 | 140 | 141 | 125 | 031 | 016 | 032 | 024 | 024 |
| T7 (232×205) | 031 | 047 | 047 | 094 | 047 | 093 | 047 | 035 | 032 |
| T8 ( 267×179) | 031 | 063 | 109 | 094 | 047 | 093 | 031 | 024 | 031 |
| T9 (480×512) | 125 | 125 | 125 | 250 | 147 | 265 | 187 | 139 | 172 |
| T10 (256×256) | 063 | 141 | 141 | 125 | 062 | 063 | 047 | 034 | 046 |
| T11 (256×256) | 062 | 062 | 141 | 156 | 062 | 062 | 062 | 046 | 047 |
| T12 (256×256) | 047 | 078 | 141 | 109 | 062 | 063 | 062 | 046 | 047 |
| T13 (256×256) | 063 | 078 | 078 | 109 | 062 | 063 | 062 | 046 | 047 |
| T14 (256×256) | 047 | 031 | 265 | 235 | 063 | 062 | 032 | 026 | 030 |
| T15 (256×256) | 047 | 031 | 063 | 125 | 078 | 062 | 062 | 046 | 047 |

### 5.2.4. Conclusions

In this study, we developed a technique by combining fuzzy logic and mathematical morphology for producing optimal edges specifically for cephalometric images. The preliminary results obtained for cephalometric images are promising, and better than both Canny and Susan. The method requires only one user defined parameter (edge detail parameter, $Th$), that needs to be adjusted coarsely according to the input image, and the details required from coarse to fine.

## 5.3. Subpixel Edge Detection

Subpixel edge detection helps to find the exact location of edges within a pixel in an image and thereby improve the precision of computer vision applications like automatic cephalometric landmark detection. Most subpixel level edge detection methods proposed in literature are based on Ghosal and Mehrotra's method [83], which uses Zernike moments. However, PZM has proven to be superior to Zernike moments in terms of their feature representation capabilities and sensitivity to image noise. In this section, we propose PZM for subpixel edge detection.

### 5.3.1. Proposed Subpixel Edge Detection using Pseudo-Zernike Moments

Considering step edge pattern $h$ is the edge height, $b$ is the background intensity, $\ell$ is the distance between the centre of unit circle, and the edge and $\beta$ is the direction of the edge. These four parameters $(h, b, \ell, \beta)$ can be calculated using Pseudo-Zernike

polynomials of different orders. For finding edge parameters Pseudo-Zernike masks, $PZ_{00}$ , $PZ_{10}$, $PZ_{11}$, and $PZ_{20}$ are calculated using orthogonal complex polynomials

$$V_{00} = 1 \tag{5.14}$$

$$V_{10} = -2 + 3r \tag{5.15}$$

$$V_{11} = r \tag{5.16}$$

$$V_{20} = 3 + 10r^2 - 12r \tag{5.17}$$

Simplifying $R_{20}$ and $R_{10}$ we obtain the following equation that will be used to find $\ell$ and $h$ parameters.

$$V_{20} - 4R_{10} = -5 + 10r^2 \tag{5.18}$$

If we rotate the edge by an angle $-\beta$ it will be aligned parallel to the $y$ axis. So we have

$$\iint\limits_{x^2+y^2\leq 1} f'(x,y)ydydx = 0 \tag{5.19}$$

$f'(x,y)$ is the edge function after it is rotated. The corresponding moments $PZ_{pq}$ of the original image and $PZ'_{pq}$ of the rotated image are related by $PZ'_{00} = PZ_{00}$, $PZ'_{10} = PZ_{10}$, $PZ'_{20} = PZ_{20}$ *and* $PZ'_{11} = PZ_{11}exp\,(j\phi)$. After the edge is rotated through an angle $\beta$, the imaginary part of $PZ_{11}$ is zero while the edge is parallel with $y$ axis, so the rotation angle of the edge is

$$\beta = tan^{-1}\left(\frac{Im[PZ_{11}]}{Re[PZ_{11}]}\right) \tag{5.20}$$

Further the following equations can be deduced based on the theory of PZM

$$PZ'_{00} = b\pi + \frac{h\pi}{2} - hsin^{-1}\ell - h\ell\sqrt{1-\ell^2} \tag{5.21}$$

$$PZ'_{11} = \iint f'(x,y)(x-jy)dydx = \frac{2h\left(1-\ell^2\right)^{3/2}}{3} \tag{5.22}$$

$$(PZ'_{20} - 4PZ'_{10}) = \iint f'(x,y)(10x^2 + 10y^2 - 5)dydx = \frac{10h\ell(1-\ell^2)^{3/2}}{3} \tag{5.23}$$

Solving Eq. (5.22) and Eq. (5.23) the edge parameter $\ell$ can be deduced as

$$\ell = \frac{PZ'_{20} - 4PZ'_{10}}{PZ'_{11}} \tag{5.24}$$

Criterion $h$ and $k$ is obtained by solving Eq. (5.21) and Eq. (5.22) respectively.

$$b = \frac{PZ_{00} - h\pi/2 + hsin^{-1}(\ell) + h\ell\sqrt{(1-\ell^2)}}{\pi} \tag{5.25}$$

$$h = \frac{1.5PZ'_{11}}{(1-\ell^2)^{3/2}} \tag{5.26}$$

Masks of any desired size can be obtained by evaluating the associated PZM integral over each pixel assuming $f(x,y)$ to be constant over that pixel. If we consider $f(x,y)=1$ then the mask coefficients can be calculated using the following equation.

$$PZ_{pq} = \frac{p+1}{\pi} \iint V_{pq}^{*}(p,\beta)dxdy \tag{5.27}$$

In the proposed algorithm five PZM masks $PZ_{00}$, $PZ_{11}$ (real and imaginary), $PZ_{10}$ (real) and $PZ_{20}$ (real) are used to calculate the edge parameters (Fig 5.7).

**General Steps to Calculate Edges Based on Pseudo-Zernike Moments**

i. Masks corresponding to different orders of Pseudo-Zernike polynomial and of desired size are calculated using circular limits of integration. In the present case, the selected masks are of size $5 \times 5$.

| | | | | |
|---|---|---|---|---|
| 0.1600 | 0.1600 | 0.1600 | 0.1600 | 0.1600 |
| 0.1600 | 0.1600 | 0.1600 | 0.1600 | 0.1600 |
| 0.1600 | 0.1600 | 0.1600 | 0.1600 | 0.1600 |
| 0.1600 | 0.1600 | 0.1600 | 0.1600 | 0.1600 |
| 0.1600 | 0.1600 | 0.1600 | 0.1600 | 0.1600 |

(a)

| | | | | |
|---|---|---|---|---|
| 0.0452 | 0.0226 | 0.0 | -0.0226 | -0.0452 |
| 0.0453 | 0.0226 | 0.0 | -0.0226 | -0.0453 |
| 0.0452 | 0.0226 | 0.0 | -0.0226 | -0.0452 |
| 0.0453 | 0.0226 | 0.0 | -0.0226 | -0.0453 |
| 0.0452 | 0.0226 | 0.0 | -0.0226 | -0.0452 |

(b)

| | | | | |
|---|---|---|---|---|
| 0.0452 | 0.0453 | 0.0452 | 0.0453 | 0.0452 |
| 0.0226 | 0.0226 | 0.0226 | 0.0226 | 0.0226 |
| 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| -0.0226 | -0.0226 | -0.0226 | -0.0226 | -0.0226 |
| -0.0452 | -0.0453 | -0.0452 | -0.0453 | -0.0452 |

(c)

| | | | | |
|---|---|---|---|---|
| 0.0330 | -0.0069 | -0.0228 | -0.0069 | 0.0330 |
| -0.0069 | -0.0620 | -0.0892 | -0.0620 | -0.0069 |
| -0.0288 | -0.0892 | -0.1340 | -0.0892 | -0.0288 |
| -0.0069 | -0.0620 | -0.0892 | -0.0620 | -0.0069 |
| 0.0330 | -0.0069 | -0.0228 | -0.0069 | 0.0330 |

(d)

| | | | | |
|---|---|---|---|---|
| -0.0093 | -0.0416 | -0.0421 | -0.0416 | -0.0093 |
| -0.0416 | -0.0133 | 0.0313 | -0.0133 | -0.0416 |
| -0.0421 | 0.0313 | 0.1467 | 0.0313 | -0.0421 |
| -0.0416 | -0.0133 | 0.0313 | -0.0133 | -0.0416 |
| -0.0093 | -0.0416 | -0.0421 | -0.0416 | -0.0093 |

(e)

Figure 5.7: Pseudo-Zernike moment masks (a)$PZ_{00}$ , (b), $PZ_{11R}$ (c) $PZ_{11I}$, (d) $PZ_{10}$, and (e) $PZ_{20}$.

ii. The masks are convolved with the image points to get Pseudo-Zernike moments $PZ_{00}$, $PZ_{11R}$, $PZ_{11I}$, $PZ_{10}$ ,& $PZ_{20}$.

iii. Use these moments to generate the edge parameters $\ell$, $b$ , $h, and\ \phi$.

iv. The above steps are repeated for every point in the image plane.

v. Using edge parameter $h,\ \ell,$ and $b$ decision is made if a pixel is an edge or a nonedge pixel.

vi. Subpixel edge location are found using $\ell$ and $\beta$

vii. If $\ell/_N < \delta$ where $2\delta$ is the size of pixel, $N$ is the size of the mask

$$x_{new} = x + 1/\sqrt{2}\, N \times \ell cos(\beta)$$
$$y_{new} = y + 1/\sqrt{2}\, N \times \ell sin(\beta) \quad (5.28)$$

$x_{new}$ and $y_{new}$ are the new edge locations with subpixel accuracy.

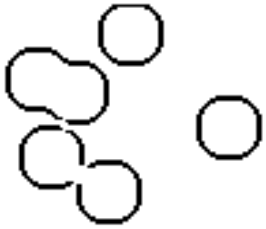
### 5.3.2. Results and discussions

The proposed algorithm was tested using several natural and synthetic gray scale images of various sizes. Both visual and quantitative results (edge thinness, missed edges, false edges, subpixel location accuracy, time complexity and behavior in presence of noise) obtained by the proposed algorithm were compared to those obtained by Ghosal et al. [82] and Bin et al. [91] method. Experimentally, it is established that the proposed algorithm gives better edge results than the Ghosal's Zernike moment based method and Bin's Fourier Mellin moments. The edge results for Ghosal's method are given in Fig. 5.8 (c) and Fig. 5.8 (d), Bin's method results are shown in Fig. 5.8 (e), and Fig. 5.8 (f) and the proposed algorithm in Fig. 5.8 (g) and Fig. 5.8 (h) for two synthetic test images shown in Fig 5.8 (a) and (b). From the visual analysis of the results it becomes clear that the edges given by the proposed algorithm are smoother, with fewer false edges than Ghosal's method for same threshold value taken for $k$ and $\ell$. All algorithms miss certain edges for values of $\ell$ in the range of $-\delta$ and $\delta$, the boundaries of the central pixel for mask size of $5 \times 5$. The proposed algorithm misses very few edges points compared to the other two methods. Ghosal's and Bin's methods do not detect corner edges accurately whereas the proposed algorithm gives better results for the corners. The only drawback of the proposed algorithm is that the edge thickness is slightly higher than the Ghosal's method.

(a) (b)

© for$k = 10$ and$\ell < \pm 0.2$ (d) for $k = 15$ and$\ell < \pm 0.2$

(e) for $k = 150$ and$\ell < \pm 0.2$ (f) for $k = 50$ and$\ell < \pm 0.2$



(g) for $k = 10$ and $\ell < \pm 0.2$ (h) for $k = 10$ and $\ell < \pm 0.2$

Figure 5.8: Two test images used for testing the proposed algorithm (a) and (b). The results for Ghosal and Mehrotra's method are given in (c) and (d), Bin's method is given in (e) and (f), and Proposed algorithm in (g) and (h).

The subpixel location accuracy for all the three algorithms is given in Table 5.2. Test images with step edges were considered for finding the subpixel locations of the edge pixels. The results show that the proposed algorithm and Bin's method give comparable results in terms of location accuracy, which is better than Ghosal's method. Further enhancement of methods and experimentation is required for other types of edges like ramp and roof.

**Table 5.2: Subpixel locations corresponding to the proposed algorithm, Ghosal's method and Bin's method.**

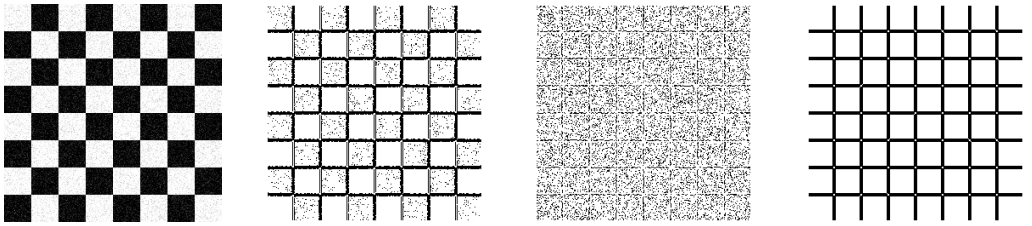
| Test Image | Actual Edge Location | | Ghosal's method | | Bin's Method | | Proposed algorithm | |
|---|---|---|---|---|---|---|---|---|
| | x | y | x | y | x | y | x | y |
| $T_1$ | 02.0 | 48.0 | 02.0000 | 47.9987 | 01.9000 | 47.9987 | 02.0000 | 47.9650 |
| $T_2$ | 48.0 | 49.0 | 47.9600 | 49.1600 | 48.0140 | 49.0137 | 47.9878 | 49.0110 |
| $T_3$ | 63.0 | 09.0 | 63.0359 | 08.8143 | 62.9989 | 09.0072 | 62.9780 | 09.0219 |
| $T_4$ | 51.0 | 03.0 | 50.9780 | 03.1134 | 50.9821 | 03.0002 | 50.9898 | 03.0000 |

The time complexity of the three algorithms is analyzed. All the three methods were implemented in C and executed on PC with 2.8 GHz CPU and 1 GB RAM. The time taken for ten test images is shown in Table 5.3. The programs were run ten times for each test image, and the average time was taken. Overall Zernike method is most time effective and Fourier Mellin the least time efficient. The proposed algorithm on average takes 40% additionally time than Ghosal's method and Bin's method takes 50% longer time than the proposed algorithm.

The results of the three algorithms in presence of Gaussian noise are shown in Fig. 5.9. The performance of Ghosal's method is poor in presence of noise. Bin's method gives better results than Ghosal's method. The results of the proposed algorithm are extremely good as is evident from Fig. 5.9 (d).

**Table 5.3: Table shows time complexity of the three algorithms in milliseconds.**

| Test Image Considered | Image Size in pixels | Ghosal's method | Bin's method | Proposed algorithm |
|---|---|---|---|---|
| T_ Img1.tif | 100X100 | 16 | 31 | 16 |
| T_ Img2.tif | 164X164 | 16 | 62 | 39 |
| T_ Img3.tif | 200X200 | 32 | 94 | 63 |
| T_ Img4.tif | 256X256 | 63 | 187 | 94 |
| T_ Img5.tif | 256X256 | 47 | 156 | 94 |
| T_ Img6.tif | 254X298 | 62 | 188 | 109 |
| T_ Img7.tif | 254X298 | 62 | 219 | 110 |
| T_ Img8.tif | 457X357 | 156 | 484 | 265 |
| T_ Img9.tif | 472X467 | 219 | 641 | 314 |
| T_ Img10.tif | 500X362 | 172 | 485 | 282 |



(a)$\sigma = 0.02$ (b) Bin's Method (c) Ghosal's Method (d) Proposed algorithm

Figure 5.9: The edge results for the three algorithms for noisy images containing Gaussian noise.

### 5.3.3 Conclusions

Pseudo-Zernike moments have been proven to be superior to other moment functions such as Zernike moments in terms of their feature representation capabilities and sensitivity to image noise. This study investigates and proposes a technique based on Pseudo-Zernike moment for pixel level and subpixel level edge detection that gives improved results than Zernike moment based method proposed by Ghosal et al. [83], and Fourier Mellin moment based method proposed by Bin et al. [89]. A square to circular mapping technique is used where all pixels of a discrete square image are mapped inside the unit circle. The proposed algorithm gives very few false edges, the edges obtained are smooth and misses few desired edge points. The performance of the method in terms of subpixel location accuracy is good. The result in presence of additive noise like Gaussian is significantly better than Zernike moment, and Fourier Mellin moment based method. The only drawback of the proposed algorithm is edge thickness. It gives thicker edges than Zernike based methods.