Tissue Segmentation Using Medical Image Processing Chain Optimization

By

Zaid Mohamad

A Thesis Submitted in Partial Fulfillment
of the Requirements for the Degree of
Master of Applied Science

The Faculty of Engineering and Applied Science
Electrical and Computer Engineering
University of Ontario Institute of Technology
August 2016

Abstract

Surveyed literature shows many segmentation algorithms using different types of optimization methods. These methods were used to minimize or maximize objective functions of entropy, similarity, clustering, contour, or thresholding. These specially developed functions target specific feature or step in the presented segmentation algorithms. To the best of our knowledge, this thesis is the first research work that uses an optimizer to build and optimize parameters of a full sequence of image processing chain.

This thesis presents a universal algorithm that uses three images and their corresponding gold images to train the framework. The optimization algorithm explores the search space for the best sequence of the image processing chain to segment the targeted feature. Experiments indicate that using differential evolution to build Image processing chain (IPC) out of forty-five algorithms increases the segmentation performance of basic thresholding algorithms ranging from 2% to 78%.

Solving the mentioned problem requires a bi-level optimization algorithm with the outer optimizer to build an optimal sequence of algorithms forming the Image Processing Chain (IPC). The inner optimization algorithm uses a chain built by the outer optimizer to optimize corresponding parameters utilized in the IPC. Using optimizer algorithms in such a configuration requires running DE optimization for every population member of the outer optimizer to fine tune the parameters needed by the population member.

The proposed framework is tested on a comprehensive set of X-ray Lung and Ultrasound Breast images; results are promising in terms of precision, sensitivity, and overlapping metrics. This framework can be utilized not only for tissue segmentation in medical images, but also for accomplishment of any complex image processing task in other areas of science and technologies.

Contents

.... i

Abstract i

1 Introduction 1

1.1 Motivation 2

1.2 Objective and Scope 2

1.3 Outline of the Thesis 4

2 Background 5

2.1 Differential Evolution 6

2.1.1 Differential Evolution (DE) 6

2.1.2 Population initialization 6

2.1.3 Mutation 6

2.1.4 Selection 7

2.2 Summary 10

3 Literature Review 11

3.1 Ant Colony Optimization (ACO): 12

3.2 Artificial Bee Colony (ABC) 13

3.3 Root Foraging Behavior (AFRO): 13

3.4 Micro Bats Algorithm (BA) 13

3.5 Cuckoo Search algorithm 14

3.6 Differential Evolution (DE) Algorithms 14

3.7 Dynamic Programming (DP) Algorithms 14

3.8 Electro Magnetism Optimization (EMO) Algorithm 16

3.9 Evolutionary Programming (EP)algorithm 16

3.10 Firefly algorithm (FA) 16

3.11 Shuffled Frog Leaping algorithm (SFLA) 17

3.12 Genetic Algorithm (GA) 17

3.13 Summary 17

4 Proposed Image Processing Algorithms 19

4.1 Image Processing 20

4.2 Image Segmentation 20

4.2.1 Edge Detection Techniques 20

4.2.2 Thresholding 22

4.2.3 Region-Based Segmentation 22
4.2.4 Watershed Segmentation 23
4.1 Filtering Algorithms: 23
4.1.1 Median Filter: 23
4.1.2 Stick Filter: 24
4.1.3 Unsharp Masking Filter: 26
4.1.4 Wiener: 27
4.1.5 Average Filter: 28
4.1.6 Disk Filter 29
4.1.7 Gaussian Filter: 30
4.1.8 Laplacian of Gaussian (LoG) Filter: 32
4.1.9 Laplacian Filter: 34
4.1.10 Prewitt Filter: 35
4.1.11 Sobel 37
4.1.12 Morphological filters: 38
4.2 Thresholding algorithms 44
4.2.1 Otsu 44
4.2.2 Kittler (Minimum Error Method) 45
4.2.3 Minimum cross-entropy threshold selection 46
4.2.4 Triangle method: 47
4.3 Post Processing Filters 48
4.3.1 Hole Filling 48
4.3.2 Erosion 49
4.3.3 Dilation 50
4.3.4 Opening 50
4.3.5 Closing 51
4.4 Contrast Enhancers 52
4.4.1 Contrast stretching (Global): 52
4.4.2 Histogram Equalization 53
4.4.3 Adaptive Histogram Equalization (local): 54
4.4.4 Fuzzy Enhancement 55
4.5 Algorithms parameters limits 58
4.6 Summary 60

5 Image Processing Chain Framework 61
  5.1 Proposed Image Processing Chain (IPC) 62
  5.2 Building Image Processing Chain Chromosome 64
  5.3 Building Algorithm Pools 66
  5.4 DE parameter Settings 67
  5.5 Image Processing Chain parameters’ indexes 68
  5.6 Summary 70
6 Case Studies and Discussions 71
  6.1 Case Study: Segmenting Lungs X-Ray images 72
    6.1.1 Training Phase 72
    6.1.2 Optimal Image Processing Chain 76
    6.1.3 Optimal Parameters 76
    6.1.4 Testing Phase 77
    6.1.5 Results analysis 78
  6.2 Case Study: Segmenting of Breast Cancer Ultrasound images 79
    6.2.1 Training Phase 79
    6.2.2 Optimal Image Processing Chain 83
    6.2.3 Optimal Parameters 83
    6.2.4 Testing Phase 84
    6.2.5 Result Analysis 85
7 Conclusion and Future work 86
  7.1 Conclusion 87
  7.2 Future work 87
8 Appendices 97
  8.1 Appendix A 98
  8.2 Appendix B 104
  8.3 Appendix C 105
  8.4 Appendix D 111

Figures

Figure 1-1: Graphical presentation of learning platform 3
Figure 1-2: Graphical flowchart of segmentation process using learned IPC 3
Figure 2-1: DE algorithm Flow Chart 8
Figure 4-1: Segmented lung 20
Figure 4-2: Edge models used in edge detection 21
Figure 4-3: Different edge detector operator 21
Figure 4-4: Laplacian mask 21
Figure 4-5: Median filter replaces the processed pixel with the median of neighboring pixels' (3x3) window size 24
Figure 4-6: Median filter results with different window sizes 24
Figure 4-7: Different angles used in stick filter 25
Figure 4-8: Effect of increasing stick filter matrix size 26
Figure -4-9: Effect of Unsharp Masking 27
Figure 4-10: Applying Wiener filter with different window sizes 28
Figure 4-11: Averaging Masks with different sizes 28
Figure 4-12: Effect of averaging mask windows size when applied on Lung X-Ray image 29
Figure 4-13: Disk shape filter mask with different sizes 29
Figure 4-14: Effect of disk averaging filter with increasing radius size 30
Figure 4-15: Two sizes of Gaussian filter masks (3x3) and (5x5), sigma 0.7 30
Figure 4-16: Effect of Gaussian filter with increasing windows size 31
Figure 4-17: Gaussian filter graph with 100x100 window size, sigma = 3 31
Figure 44-18: Gaussian bell windows' size = 100, sigma = 20 32
Figure 4-19: Effect of Gaussian filter with increasing sigma 32
Figure 4-20: Effect of LoG filter with fixed window size [20 400] and increasing sigma value 33
Figure 4-21 Showing pixel information contained in black lung X-Ray image filtered with LoG with sigma = 1 33
Figure 4-22: Effect of Laplacian filter applied on moon image 34
Figure 4-23: Gx,y Masks representing [4-9,10] equation as matrices 35
Figure 4-24: First row show Prewitt output when applied on Lung X-Ray image (black images), second row of images is the result of adding Prewitt output to original image to emphasis edges 36
Figure 4-25: Applying histogram equalization on Prewitt filter 36
Figure 4-26: Effect of applying Sobel filter on Lung X-Ray image 37
Figure 4-27: Histogram Equalization applied on sobel filtered images 38
Figure 4-28: (a) Diamond (b) Disk (C) Octagon Structural Elements 39
Figure 4-29: Different shapes of structural elements 39
Figure 4-30: Different shapes of structural elements 39
Figure 4-31: Effect of Erosion applied on Lung X-Ray image 41
Figure 4-32: Magnification of zone 1 in fig. 4-31 showing the effect of erosion 41
Figure 4-33: Effect of Dilation with three-pixel diameter disk shape SE 42
Figure 4-34: Effect of opening with disk shape SE 43
Figure 4-35: Effect of closing with disk shape SE 43
Figure 4-36: Effect of Otsu Thresholding 45
Figure 4-37: Kittler Thresholding 46
Figure 4-38: Minimum Cross-Entropy thresholding 47
Figure 4-39: Triangle thresholding technique 48
Figure 4-40: Triangle thresholding 48
Figure 4-41: Hole Filling effect 49

Figure 4-42: Effect of eroding with 3x3 structure element 49
Figure 4-43: Effect of dilation by 3x3 structure element 50
Figure 4-44: Effect of applying opening 51
Figure 4-45: Morphological opening illustrated as rolling ball on the inside border of image 51
Figure 4-46: Morphological closing illustrated as rolling ball on the outside surface of image 52
Figure 4-47: Gamma transformation function with different gamma values 53
Figure 4-48: Effect of contrast stretching 53
Figure 4-49: Illustration of histogram equalization effect on image histogram 54
Figure 4-50: Effect of local histogram equalization 55
Figure 4-51: Original image (left), lambda enhancement result (right) 57
Figure 4-52: Original (left) and enhanced image (right) using fuzzy intensification enhancer 58
Figure 5-1: Illustration of Image Optimization Chain and algorithms used by the DE optimizer 64
Figure 5-2: DE built IPC segmentation sequence 66
Figure 6-1: Three training images with their respective gold images 72
Figure 6-2: Program GUI interface during training phase 73
Figure 6-3: Segmentation results by IPC during generations 2,63, and 162 74
Figure 6-4: Segmentation results by IPC during generations 365,423,620,1349, and 2220 75
Figure 6-5: Fitness value vs Generation 75
Figure 6-6: Segmentation using trained IPC 78
Figure 6-7: Three training images for breast cancer 79
Figure 6-8: Segmentation results for 2nd and 12th generation 80
Figure 6-9: Segmentation results for 28th,29th,76th, and 101 generations 81
Figure 6-10: Segmentation results for 125th,410th,705th, and 1297 generation 81
Figure 6-11: Segmentation results for 1677th generation 82
Figure 6-12: Fitness vs Generation 82

Tables

Table 4-1: Minimum and Maximum for Filters’ parameters allowed values 58
Table 4-2: Minimum and Maximum Values for Contrast Enhancer algorithms’ parameters 59
Table 4-3: Minimum and Maximum values for Thresholding algorithms parameters 59
Table 4-4: Minimum and Maximum for Post Processing Algorithms parameters...59
Table 5-1: Differential Evolution optimizer parameter settings...67
Table 5-2: Filters parameters location in the chromosome (Index) 69
Table 5-3: Contrast enhancers parameters location in the chromosome (Index)..69
Table 5-4: Post Processing algorithms’ parameters’ location (index) 70
Table 6-1: DE parameter settings for lung segmentation 73
Table 6-2: Image Processing Chain as created by DE...76
Table 6-3: Algorithms’ parameters stored in optimized chromosome indexes 11-18 77
Table 6-4: Algorithms’ parameters stored in optimized chromosome indexes 19-26 77
Table 6-5: Algorithms’ parameters stored in optimized chromosome indexes 27-33 77
Table 6-6: Algorithms’ parameters stored in optimized chromosome indexes 34-40 77
Table 6-7: Algorithms’ parameters stored in optimized chromosome indexes 41-48 77
Table 6-8: Algorithms’ parameters stored in optimized chromosome indexes 49-56 77
Table 6-9: Algorithms’ parameters stored in optimized chromosome indexes 57-64 77
Table 6-10: Segmenting images using individual thresholding algorithms used in IPC 79
Table 6-11: DE parameter settings for Lung segmentation...80
Table 6-12: Image Processing Chain as created by DE 83
Table 6-13: Algorithms’ parameters stored in optimized chromosome indexes 11-18 84
Table 6-14: Algorithms’ parameters stored in optimized chromosome indexes 19-26 84
Table 6-15: Algorithms’ parameters stored in optimized chromosome indexes 27-33 84
Table 6-16: Algorithms’ parameters stored in optimized chromosome indexes 34-40 84
Table 6-17: Algorithms’ parameters stored in optimized chromosome indexes 41-48 84
Table 6-18: Algorithms’ parameters stored in optimized chromosome indexes 49-56 84
Table 6-19: Algorithms’ parameters stored in optimized chromosome indexes 57-64 84
Table 6-20: Segmenting images using individual thresholding algorithms used in IPC 85

Chapter 1

Introduction

1.1 Motivation

Medical imaging technology has revolutionized health care over the last thirty years, allowing doctors to diagnose diseases earlier and target illnesses before they become untreatable.

Accurate applied segmentation on these medical images to isolate specific tissues or organs is considered as a critical task to make the correct diagnoses of diseases. These segmentation techniques are the main tools in the medical sector, providing early and effective diagnoses.

Current segmentation techniques of medical images are used for some applications such as imaging modality and the especial study of body part. Many factors should be considered such as sensors noise, tissue motion artifacts, ring artifacts, etc.

Each imaging system has its own specific limitations and produces images with different challenges for the segmentation system; therefore, it is of interest of this thesis to develop a universal algorithm for segmentation of every medical image which according to the best of knowledge, it has not done so far.

Expert systems use multiple techniques such as statistical models or artificial intelligence which require the huge databases of training samples; so they are expensive approaches, and in some cases impractical.

In this thesis, we consider the above mentioned problems of imaging systems and the impracticality of building expert systems. A universal system is developed which is trained to segment tissues using three training images from specific imaging modality. The system generates a trained image processing chain which is loaded to segment images produced by the same imaging modality.

1.2 Objective and Scope

This thesis proposes a universal trainable framework that can target any image processing task for any image modality. This universal framework can target industrial, medical, satellite, etc. or any problem that require image processing. The proposed framework can build proper image processing chain that target specific problem based on provided gold samples.

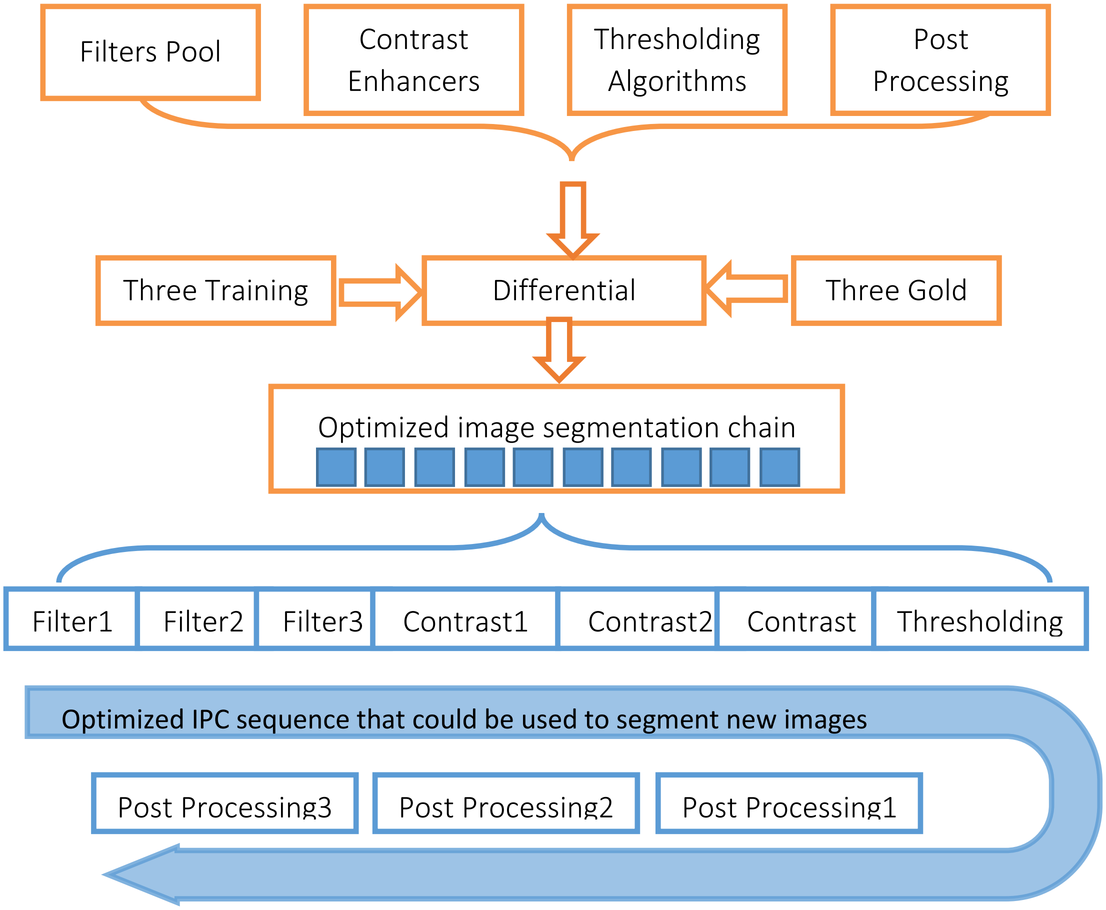


Figure 1-2: Graphical presentation of learning platform

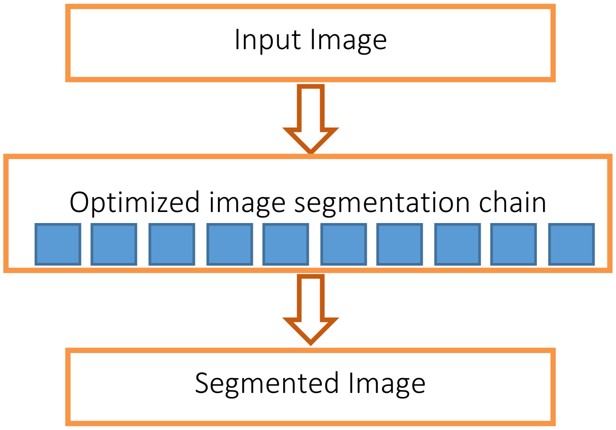


Figure 1-1: Graphical flowchart of segmentation process using learned IPC

In this thesis we use it in medical image segmentation to extract specific tissues. This framework as illustrated in figure (1-1/2) uses three training images to be trained on specific tissue in images generated by specific imaging modality.

The scope of this thesis can be summarized in two areas listed below which determine how we can achieve the mentioned goals:

1- Develop an optimizable image processing chain that is trained by three images to segment targeted tissue.
2- Use the differential evolution algorithm to find the optimal series of filters to get the segmentation as close as possible to physician segmentation.

1.3 Outline of the Thesis

This thesis is organized into seven chapters. Chapter 2 presents a background review on image processing techniques, namely, image enhancement, thresholding, mathematical morphology, and object segmentation. It also discusses differential evolution algorithm that is utilized in this thesis.

In Chapter 3, we provide a literature review of optimization based segmentation algorithms.

Chapter 4 explains the image processing algorithms (filter, enhancers, thresholding, and post processing filters) that is used to build the proposed image processing chain; then, we move to Chapter 5, where we explain in detail the chromosome design and the fitness function which is used to optimize the chain by the DE algorithm.

In Chapter 6, we discuss two cases, namely, Lung X-Ray and Breast Cancer segmentation.

Chapter 7, provides a summary, conclusion and future work.

Chapter 2

Background

2.1 Differential Evolution

Differential Evolution (DE) is, a vector population based stochastic optimization method which was introduced by Price and Storn in 1995 [1]–[6].

In the following sections, the major definitions and mathematical formulas of the DE algorithm is described.

2.1.1 Differential Evolution (DE)

DE is a population-based stochastic method for global optimization. Population is expressed with its conditions and dimension as follows:

$$P_{x,g} = \left(x_{i,g}\right), \quad i = 0,1 \ldots \ldots, N_p - 1 \qquad g_{max} = 0,1, \ldots \ldots, g_{max} \qquad \text{eq(2-1)}$$

$$x_{i,g} = \left(x_{j,i,g}\right), \quad j = 0,1, \ldots D - 1$$

Where Np is the number of population vectors and D is the dimension of the problem (number of parameters).

2.1.2 Population initialization

DE algorithm generates randomly an initialization population. A solution vector in the initialization population is generated as follows:

$$x_{j,i,0} = rand_j[0\ 1].\left(b_{j,U} - b_{j,L}\right) + \ b_{j,L} \qquad (2\text{-}2)$$

Where $b_L and\ b_U$ are a D-dimension lower and upper limit vector and,

$rand_j[0\ 1]$ is a uniform random number in the interval [0 ,1].

The index 0 indicates that this is generation zero (initialization phase)

2.1.3 Mutation

The DE optimizer randomly chooses three vectors for each target vector in the population. These three vectors are being used to generate a mutated vector as follows:

$$v_{i,g} = \ y_{i,g} + F.\left(x_{r1,g} - x_{r2,g}\right) \qquad (2\text{-}3)$$

Where$v_{i,g}$ is the mutated vector v_i for generation g and$F$ weight that is applied to the vector difference $\left(x_{r1,g} - x_{r2,g}\right)$, and $y_{i,g}$ is the base vector.

. Other mutation strategies are as follows:

Classical DE/rand/1/bin

$$v_{i,g} = x_{r0,g} + F.(x_{r1,g} - x_{r2,g}) \tag{2-4}$$

DE/best/1/bin

$$v_{i,g} = x_{best,g} + F.(x_{r1,g} - x_{r2,g}) \tag{2-5}$$

DE/Current-to-best/1/bin

$$v_{i,g} = x_{i,g} + F.\left(x_{best,g} - x_{i,g}\right) + F.\left(x_{r1,g} - x_{r2,g}\right) \tag{2-6}$$

DE/best/2/bin

$$v_{i,g} = x_{best,g} + F.(x_{r1,g} - x_{r2,g} + x_{r3,g} - x_{r4,g}) \tag{2-7}$$

The notation for DE is defined by DE/x/y/z, where x denotes the base vector, y denotes the number of difference vectors used, and z represents the crossover method.

For example, DE/best/1/bin represents using the best vector with the lowest objective function so far to generate the mutated vector.

2.1.3.1 Crossover

DE algorithm uses the crossover operation to diversify the solutions. Based on the crossover value, the DE combines the mutated vector $v_{i,g}$ with the target vector in order to generate a trial vector.

2.1.4 Selection

Trail vector and target vector are compared and the vector with the lowest objective function value takes the place of the target vector.

Therefore, DE either keep the target vector or replace it with the trial vector in the next generation.

Figure [2-1] shows the DE flowchart in detail, along with its internal working and all components. The pseudo code of the DE algorithm is described as below:

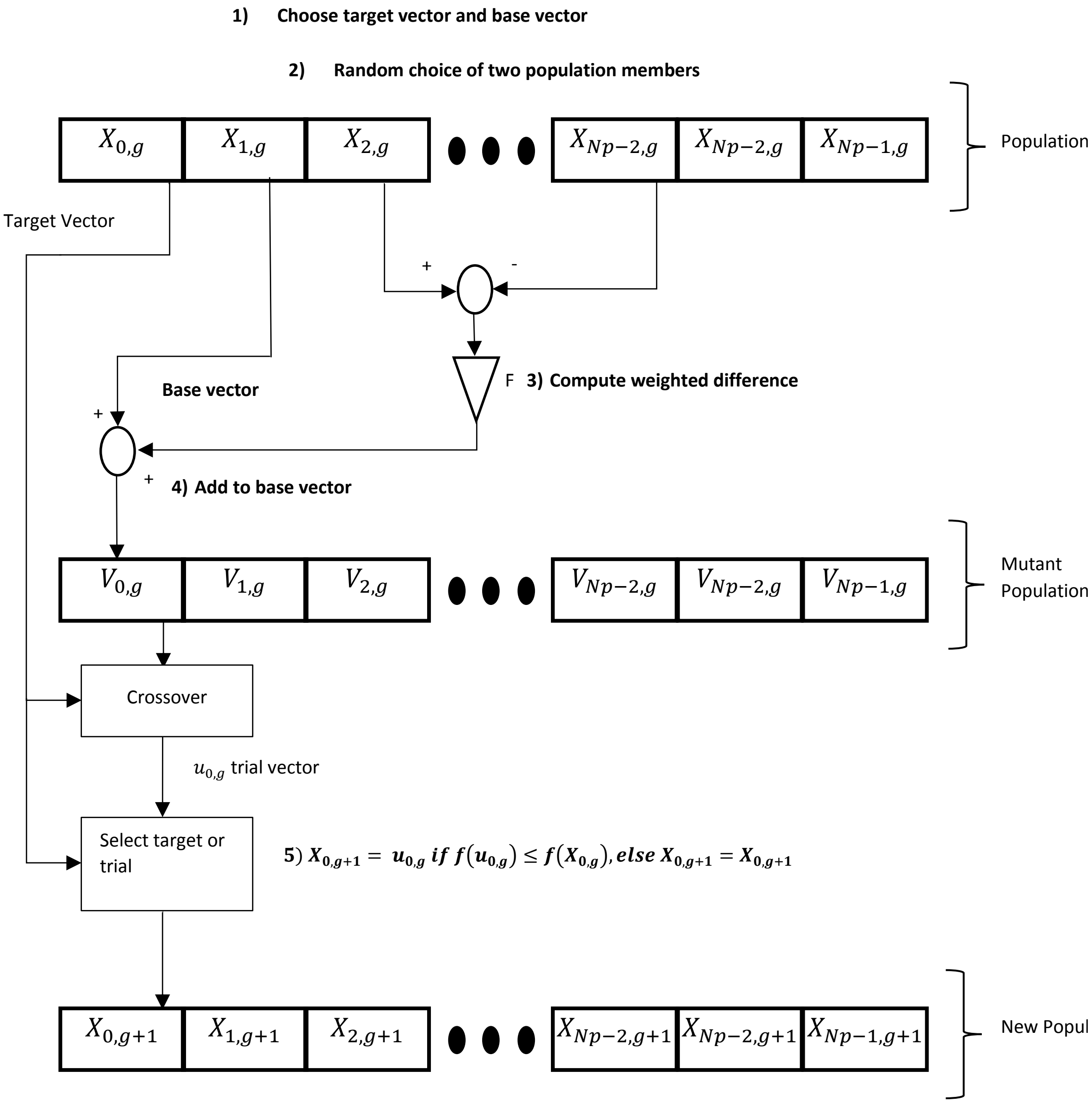


Figure 2-1: DE algorithm Flow Chart

```
Generate uniformly distributed random population $P_0$
while (BFV>VTR and NFC<MAXNFC) do
// Generate-and-Test-Loop
for i=0 to Np do
    Select three parents $X_a$, $X_b$, and $X_c$ randomly from current population where i≠a≠b≠c
```

```
// Mutation Section
        $V_i \leftarrow X_a + F * (X_c - X_b)$
        for j= 0 to D do
// Crossover Section
            if rand(0,1)<$C_r$ then
                $U_{i,j} \leftarrow V_{i,j}$
            else
                $U_{i,j} \leftarrow X_{i,j}$
            end if
        end for
// Selection Section
        Evaluate $U_i$
    if ($f(U_i) \le f(X_i)$) then
        $X'_i \leftarrow U_i$
    else
        $X'_i \leftarrow X_i$
    end if
  end for
  $X \leftarrow X'$
end while
```

Algorithm 1: Pseudo code of DE Algorithm. P_0: Initial population, Np: Population size, V: Noise vector, U: Trial vector, D: Problem dimension, BFV: Best fitness value so far, VTR: Value-to-reach, NFC: Number of function calls, MAXNFC: Maximum number of function calls, F: Mutation constant, rand (0,1): Uniformly generated random number, Cr: Crossover rate, f(.): Objective function, P': Population of the next generation

2.2 Summary

Here, we have seen the foundation principles of the DE algorithms that will be used to build and fine tune the IPC. Population initialization, mutation, crossover, and selection best population members are the basic principles that make DE works for this type of optimization

Chapter 3

Literature Review

Segmentation divides the image into regions or segments. This segmented representation of the image simplifies the process of further analysis, such as finding features, objects' boundaries, or structures. In terms of image processing, segmentation is the process of assigning a label to every pixel in the image such that pixels with the same label share the same set of attributes or characteristics.

Existing segmentation algorithms use two approaches [7]. The first approach assumes global similarity among pixels of the targeted feature that distinguish it from the pixels of the background pixels. These features could be difference in brightness value, color, or texture, which leads to methods that are concerned with finding the optimum thresholds that distinguish target features pixels. The second approach uses local features.

Other classification of segmentation algorithms is based on the similarity and discontinuity properties of intensity values in an image [8]. This thesis uses algorithms in both categories. These algorithms are pooled in a chain to be optimized by DE optimizers to find the best approach specific to imaging modality and the features which are used in the segmentation. In this chapter, we survey the state of the art of optimization-based segmentation algorithms.

The following sections classify these algorithms based on optimization techniques used in the segmentation algorithms.

3.1 Ant Colony Optimization (ACO):

Pereira et al. [9] used ACO algorithm preceded by anisotropic diffusion for optic disc detection in color fundus images. They utilized a number of moving ants on the image driven by the local variation of the image's intensity values to establish a pheromone matrix with the same size of the image which represents the edge information at each pixel location of the image. This edge enhancer was used in [10] to enhance edges as a preprocessing stage to segment exudates using 8-neighbor connected component analysis.

Wang et al. [11] used ACO to minimize the energy function of Active Contour Model to segment the left ventricle of the heart. Han et al. [12] combined Fuzzy clustering with ACO to segment images based on three extracted features, namely, intensity, gradient, and neighborhood of the pixels information abstracted in a number. Yuanjing et al. [13] proposed finite ACO combining ACO with multistage decision process of image processing based on ACM. Yu et al [14] used ACO combined with PCM to create a hybrid segmentation algorithm ACOPCM for noise images. ACO was used in the proposed algorithm to provide near optimal initialization of the number of clusters and their centroids to the clustering algorithm PCM. Biniaz et al [15] used Fuzzy c-mean (FCM) to define food to supervise ACO ants. Ants have the sense of responsibility to seek pixels with analogous properties and are able to compare the pixels and the specific reference food to select it. Liang et al [16] used ACO to search for the optimal number of thresholds to perform multilevel segmentation using Otsu thresholding of colored images.

3.2 Artificial Bee Colony (ABC)

Artificial Bee Colony optimization was also used in the image segmentation. In [17], ABC was used to find the optimal thresholding vector that maximizes a defined entropy function. These optimal thresholds were then used to segment the image into multiple sections. Taherdangkoo [18] used four groups of scout bees in ABO algorithm to identify the skull pixel to segment. ABO bees searched the whole image for skull pixels. MABC was introduced by Li et al. [19] and applied on multi-thresholding problem; this algorithm was applied to maximize an entropy criterion-based fitness measure function proposed by Otsu [20], [21]to find the best multi-thresholds to segment the provided images.

3.3 Root Foraging Behavior (AFRO):

Ma et al. [22] introduced a new optimization algorithm for multi-threshold segmentation of images inspired from plant root foraging behavior [23]. AFRO was used to maximize a function with N dimension representing N thresholds to use for the image segmentation.

3.4 Micro Bats Algorithm (BA)

Ye et al. [24] used a new metaheuristic algorithm called BA, which was inspired from micro bats that uses sonar to detect prey and avoid obstacles. They used BA to reach a thresholding value that maximizes the total fuzzy entropy functions of all classes of an image. In [25], BA is used to optimize a novel proposed objective function. BA fails to segment images when the number of thresholds is large, in [26] Improved Bat Algorithm

IBA was proposed, where they integrated DE with the crossover and mutation to efficiently explore and exploit the new search space and avoid being trapped into local optima. They also used ACO random walk to jump out of local optimum and search a new space to escape trapping in local optimum.

3.5 Cuckoo Search algorithm

Preetha et al. [27] used the Cuckoo algorithm to optimize similarity function; special similarity function was developed between the segmented image using region growing and the gold ground truth image. Suresh et al. [28] introduced an improved version of the Cuckoo search algorithm called $CS_{\mathrm{McCulloch}}$ by incorporating McCulloch's method for levy flight generation in CS algorithm. They then used $CS_{\mathrm{McCulloch}}$ to maximize Otsu's between-class variance, Kapur's entropy, and Tsallis Entropy. Bhandari et al [29] used CS to maximize Kapur's entropy to find optimum thresholds to segment satellite images. Bhandari et al [30] and Agrawal et al [31] used Tsallis entropy as a fitness function to maximize for the optimum multi threshold variables. These thresholds were used to segment colored satellite images and colored images obtained from online datasets.

3.6 Differential Evolution (DE) Algorithms

Khan et al. [32] presented new algorithms to segment colored images. They s combined the adaptive DE, where each individual in the population is assigned different CR and mutation probability, with opposite-based learning to form a new algorithm called MoADE. Sarkar et al. [33] used DE to minimize cross entropy objective function for multiple thresholds. These threshold values were used to segment the colored image. Ayala et al. [34] used beta DE algorithm (BDE) to find the optimal thresholding levels needed to segment images. BDE was used to maximize Otsu objective function. Sarkar et al. [35] used DE to maximize Rényi entropy [36] objective function for multiple thresholds. Threshold values were used to segment satellite images.

3.7 Dynamic Programming (DP) Algorithms

Liu et al [37] segmented Left Ventricles using DP. DP was used to find the global minimum of developed fitness function based on the edge map of the left ventricle LV, which is developed by using the non-maxima gradient suppression technique [38].Qi et al. [39]

suggested a new and improved DP algorithm to segmented breast masses. They developed a new cost function using edge strength, gray level and mass size features. These features were combined to compute the cumulative cost function to be minimized. The global minimum optimal is obtained to be the contour of these masses.

Maduskar et al. [40] studied the cavity contour in chest radiographs. They used pixel classifier to assign a likelihood value of belonging to the cavity border to each pixel. This likelihood map is used as an input cost image in the polar transformed image space for dynamic programming to trace the optimal maximum cost path.

Hogeweg et al. [41] presented an algorithm to segment clavicles from chest radiographs. DP was used to maximize cost function concluded from multiple classifiers targeting clavicle’s body, border, and shape pixels. Optimal path is found by maximizing this cost function. Segmenting prostate cancer is considered in [42] where discrete deformable model algorithm is used with DP to minimize an energy function. This energy function integrates three energy components related to internal contour continuity and curvature, external image feature, and partial active shape model shape guidance.

Zhou et al. [43] proposed DP to solve segmentation of carotid artery intima-media in longitudinal ultrasound images for measuring its thickness. The problem is identified as detection of parallel boundaries that complied with two features specifically, smoothness and parallelism of both boundaries. Therefore, they develop a special combined cost function to be minimized by DP.

Song et al [44] provided a segmentation algorithm that target breast lesions using DP. A template-matching technique is used to locate and obtain the rough region of masses. DP optimizer used this rough region combined with a local cost function to find the mass contour. In [45] Zahnd et al used DP optimizer to segment and quantify coronary fibrous cap thickness in optical coherence. DP minimizes a cumulative cost function specially developed for this segmentation problem.

Qian et al. [46] used DP to myocardium segmentation. DP scheme searches a globally optimal path in the polar-transformed image by minimizing the path cost consisting of both image information (i.e., external energy) and smoothness constraints (i.e., internal energy).

Rocha et al. [47] studied the segment the intima-media region. DP was used to estimate the lumen boundary as a step towards full segmentation of the intima-media region. Avila-Montes et al. [48] applied DP in the hough space for localizing key center points in the aorta slices as a step towards 3D segmentation of thoracic aorta . DP uses polar-transformed image and a cost of a path function as the sum of local energies along the path.

3.8 Electro Magnetism Optimization (EMO) Algorithm

Oliva et al [49] proposed EMO to solve multilevel segmentation of images. EMO optimizer was used to maximize Otsu between class variance or Kapur’s Entropy function. In [50], they targeted multi-level segmentation velocity of images and proposed using EMO algorithm combined with Tsallis entropy function.

3.9 Evolutionary Programming (EP)algorithm

Fang [51] proposed a new EP with two step mutation rules. New EP was used to minimize FCM function to reach optimal thresholding values for each class in the image.

3.10 Firefly algorithm (FA)

Alsmadi [52] used the firefly algorithm to search the search space to find near optimal cluster number and associated centers. This near optimal solution is then fed to FCM for the final solution. Hassanzadeh et al [53] proposed a new fuzzy-based firefly algorithm to improve explore ability. They applied it to multi-level segmentation using Otsu between variance cost function. Chen et al [54] introduced two modification on firefly algorithm specifically, diversity enhancing strategy with Cauchy mutation and neighborhood strategy. They then used the improved FA (IFA) to maximize between-class variance (Otsu function), and find the optimal multi-level thresholds for colored images.

Horng et al [55] proposed using FA to maximize the objective function of MET (maximum entropy thresholding). Multi-level thresholds are generated to segment colored images. Then, Horng et al [56] maximized the inverse of minimum cross entropy thresholding (MCET) function to obtain the optimal multi-thresholds.

3.11 Shuffled Frog Leaping algorithm (SFLA)

Ladgham et al [57] used the modified MSFLA to optimize a fitness function Both the modification of SFLA and the new fitness function aim at eliminating local optimum and increase computation time. The proposed algorithm was then applied on MR brain images for segmentation. Wang et al [58] used SFLA algorithm to maximize the trace of a derivation matrix of 3-D Otsu thresholding algorithm.

Horng [59] used SFLA to maximize a special entropy function. The optimal solution provides optimal thresholds which is used to segment the image. In [60] Horngused SFLA to minimize the cross entropy thresholder and obtain optimal threshold values for the multilevel image segmentation.

3.12 Genetic Algorithm (GA)

Khayat et al. [61] proposed a new algorithm to segment nano-scale light microscopic images. They combined two algorithms namely, gray level spatial correlation histogram and entropic criterion function [62] to calculate the best threshold of provided images. They use GA for the optimization of parameters in the combined algorithm. Xie et al. [63] proposed a new algorithm for segmenting dermoscopy images. GA algorithm is combined with between-class variance to segment the image into multiple classes; pixels from every class were seeded to a self-generating neural network.

Usamentiaga et al. [65] used GA to tune the parameters of an infrared image segmentation algorithm. In [66] Khan et al. developed a new algorithm to segment colored images. They combined spatial FCM with GA as SFGA algorithm. SFGA then is used to maximize a collective objective function. Mousavi et al. [64] proposed segmenting color images using GA. They improved segmentation accuracy by using GA to optimize color space HSI (Hue, Saturation, and Intensity).

3.13 Summary

In this chapter, we surveyed several optimization-based segmentation algorithms. The segmentation algorithms show very specific use of optimizers targeting specific task in the segmentation instead of the whole process of images segmentation (filtering, enhancing, thresholding, and post processing). These tasks can be categorized into several areas as below:

- Find edges or shortest path in polar transformed images
- Minimize or maximize energy, entropy, and similarity functions
- Extract features such as intensity, and gradient
- Initialize segmentation algorithms with clusters and their centers and optimize clustering results
- Optimize thresholding algorithms
- Maximize or minimize special functions developed specifically for the segmentation problem being investigated

Chapter 4

Proposed Image Processing Algorithms

4.1 Image Processing

Images can be represented as a two-dimensional function $f(x,y)$where the amplitude of $f$at any pair of coordinates x and y is the intensity or gray level of the image at that point.

4.2 Image Segmentation

Segmentation is partitioning the digital image into multiple segments or classes to simplify further analysis. These sets of segments/classes cover the entire image. The outputs of image segmentation are attributes extracted from those images which could be a set of extracted contours, edges, or a set of classes.

Segmentation classifies each of the pixels based on some characteristic or computed property, such as color, intensity, or texture which results in regions of similar pixels. Mathematically, it is splitting the image region R to "n" non-overlapped sub regions $R_1, \dots, R_n$ such that:

$$R = \bigcup_{i=1}^{n} R_i \qquad (4\text{-}1)$$

In this case, we are looking to segment medical images to two regions: background and foreground, representing the targeted tissue of the segmentation process. Figure 4-1 shows a sample lung X-Ray ihmage segmented into two regions.

Figure 4-1: Segmented lung

The following sections show different principles of segmenting images.

4.2.1 Edge Detection Techniques

Regions forming the segments in images are surrounded by borders with abrupt changes in intensity levels. These changes can be modeled in one of the three models (see Figure

4-2). Many segmentation techniques use these edge models to detect and contour segments of interests.

In Figure 4-2, we can see that the first model shows a step edge, which can be found in animations with one clean pixel abrupt changes in the intensity level. On the other hand, digital images have normally blurred edges which makes the detection of the ramp edge model more appropriate.

Step edge model Ramp edge model Roof edge model

Figure 4-2: Edge models used in edge detection

In ramp edge model, the changes in intensity happen over multiple pixels. The third edge model "roof edge" happens in regions of digital images where you have thin lines such as pipes or roads in satellite images.

Detecting these types of edges can be done using first order derivatives such as Roberts. Prewitt, and Sobel edge. Figure 4-3 shows the masks of these operators which can be convolved with the image to get the filtered image.

-1	0
0	1

0	1
-1	0

Roberts

-1	-1	-1
0	0	0
1	1	1

-1	0	1
-1	0	1
-1	0	1

Prewitt

-1	-2	-1
0	0	0
1	2	1

-1	0	1
-2	0	2
-1	0	1

Sobel

0	1	1
-1	0	1
-1	-1	0

-1	-1	0
-1	0	1
0	1	1

Diagonal Prewitt

0	1	2
-1	0	1
-2	-1	0

-2	-1	0
-1	0	1
0	1	2

Diagonal Sobel

Figure 4-3: Different edge detector operator

Some segmentation methods use the second derivative Laplacian operator to detect abrupt changes of the intensity. The Figure 4-4shows the Laplacian mask.

1	1	1
1	-8	1
1	1	1

Figure 4-4: Laplacian mask

Laplacian mask in Figure 4-4 is isotropic that detect lines in all directions.

Filtering images in the spatial domain by using the above masks can be represented mathematically as follow (4-2):

$$g(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t).f(x+s, y+t) \tag{4-2}$$

The above mentioned techniques normally yield discontinued pixels located on the borders, which is normally caused by a noise. Different edge linking and boundary detection are used to link these pixels to form a complete segment linked with boundary.

4.2.2 Thresholding

This technique partitions the image directly using intensity values or properties of these values or both. The basic approach is choosing a threshold value T such that all pixels $f(x,y) > T$ forms the object to be segmented, while the pixels with $f(x,y) < T$ are the background pixels. This global thresholding technique can be mathematically expressed as:

$$g(x,y) = \begin{cases} 1 \; if \; f(x,y) > T \\ 0 \; if \; f(x,y) \; \leq T \end{cases} \tag{4-3}$$

In some cases, we might be interested in multi level segmentation. In this case, we choose multiple thresholds that segment the image into some classes. The thresholded image is represented by:

$$g(x,y) = \begin{cases} a & if \; f(x,y) \leq K_1 \\ b & if \; K_1 < f(x,y) \; \leq K_2 \\ c & if \; f(x,y) > \; K_2 \end{cases} \tag{4-4}$$

4.2.3 Region-Based Segmentation

This technique groups pixels into larger regions based on the predefined criteria for growth. Seeding points of every region can be selected either manually or based on the nature of the problem. The seeds selection is followed by computing the same set of attributes for all pixels in the image. Adding pixels to a region based on predefined predicate Q is represented as follows:

$$Q = \begin{cases} TRUE & if \; the \; pixel \; at \; (x,y) \; meets \; set \; criteria \\ False & Otherwise \end{cases} \tag{4-5}$$

Another technique is using region splitting which starts by using the full image as a region and splitting the region into four disjointed regions if $Q(R_i) = False$. Splitting continues until no splitting is possible. This is followed by merging adjacent regions till no merging is possible. Merging regions is based on the predicate condition:

$$Q(R_j \cup R_k) = TRUE$$

4.2.4 Watershed Segmentation

This approach uses many concepts of the previously mentioned algorithms. It is based on dealing with the image in 3D where intensity values are considered as the Z coordinate. Using this interpretation of the image produces 3D topography with basins (forming regional minimums). We flood this topography with water from below, causing the water to rise in distinct catchment basins till it is about to merge. At this stage, we build a dam and continue the flooding process till only top of the dam is above water.

4.1 Filtering Algorithms:

Filtering is the first step in the image processing chain. The optimizer algorithm can choose up to three filters from a pool of twenty filters to build the image processing chain. Chosen filters are then applied in sequence on the image to filter noise and unwanted features and emphasize the extracted features.

4.1.1 Median Filter:

Median [67] is calculated for every pixel in an image based on the window size chosen for the filter as it is shown in the Figure 4-5. The algorithm arranges the values of pixels' intensity levels in the window surrounding the targeted pixel and identifies the middle value to replace the targeted pixel. The filter has excellent ability to reduce noise as well as preserve edge characteristics, which makes it fit for our pool of pre-processing filters in the optimization chain.

Median filter requires an especial window size which in turn requires two parameters; each of them varies between 2 to the maximum size of the image. Using high limits for these parameters causes the deterioration of the image and loss of critical data that can used to identify an accurate border of the targeted tissue.

Figure 4-6 shows how border information of lungs deteriorates as we increase windows size from $[10,10]$ to $[150,150]$. In order to maintain the border information, we have set the limit on median filter window size to $[150,150]$.

$$\hat{f}(x,y) = median\{g(s,t)\}_{(s,t)\in S_{xy}} \tag{4-6}$$

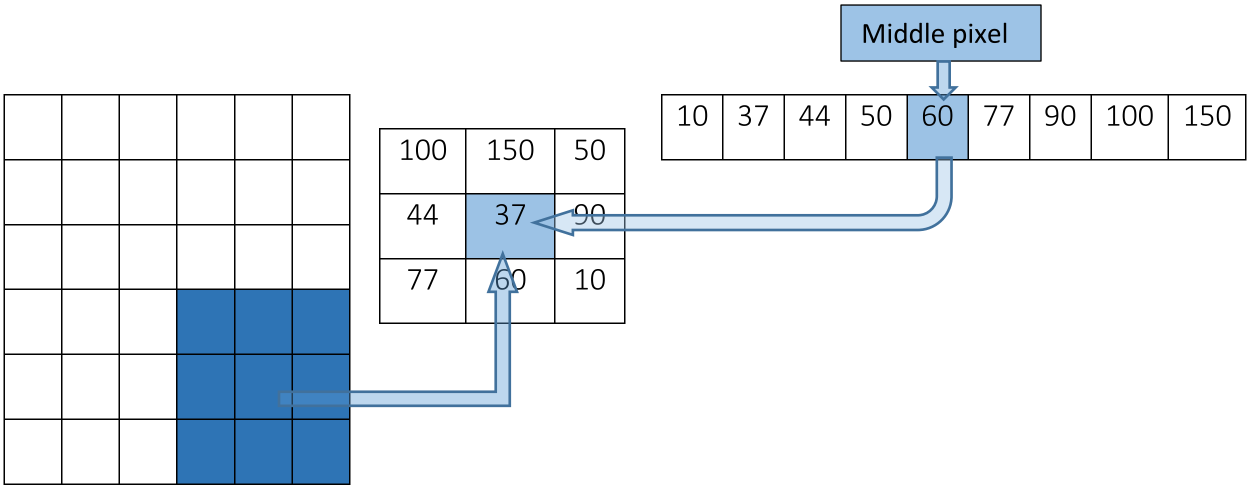


Figure 4-5: Median filter replaces the processed pixel with the median of neighboring pixels' (3x3) window size

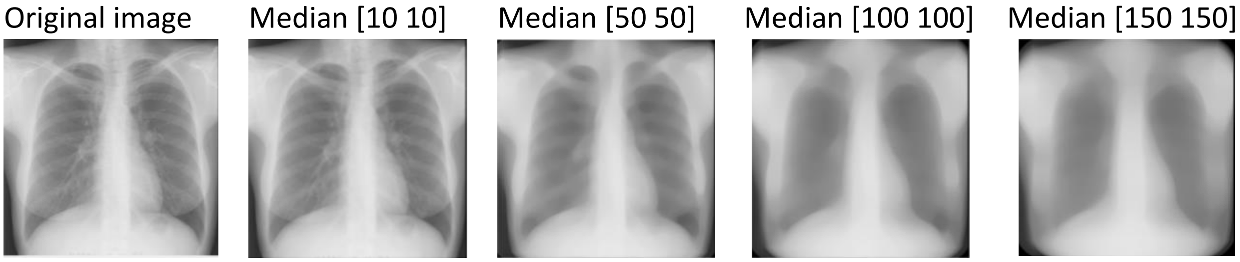


Figure 4-6: Median filter results with different window sizes

4.1.2 Stick Filter:

Stick filter [68] detects speckles by maximizing the likelihood ratio of a set of image pixels in a neighborhood and a straight line passing through the centre of the neighbourhood. Increasing "stick" length produces a smooth filtered image at the expense of weakly highlighting tightly bound curves, while thicker "sticks" suppress more noise at the expense of making thin boundaries less visible.

Parameterized stick filter function used in our optimization chain gives the chain a good tool to reduce speckles while emphasizing edges which is beneficial for the overall performance of the optimized chain.

The filter generates eight matrices representing sticks in eight different angles. Each stick has specific thickness as it can be seen from Figure 4-7 .The algorithm for the detection is based on decision-directed Sticks algorithm, which can be summarized in three steps: the determination of the most probable line direction at each point, the computation of a prior probability for each angle at each point, and the computation of the final test statistic.

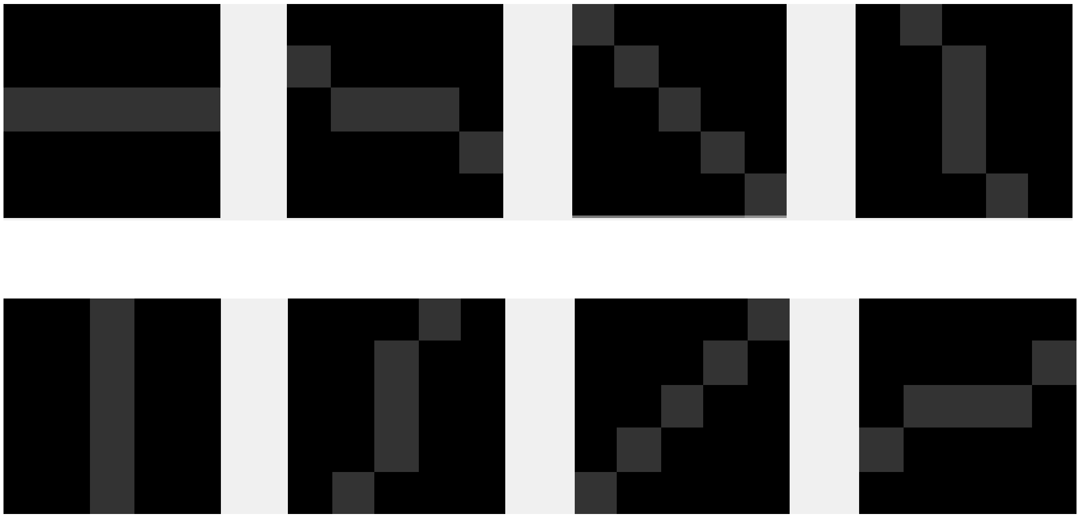


Figure 4-7: Different angles used in stick filter

The equations of this approach are described as below:

$$\propto (x,y) = \arg\max_{\theta} \sum\nolimits_{i,j} s_{\theta}(i,j)\, f(x-i, y-j) \tag{4-7}$$

$$\pi_{\theta}(x,y) = \sum_{i,j} w(x+i, y+j)\, I(\propto (x,y) = \theta) \tag{4-8}$$

$$\acute{f}(x,y) = \max_{\theta_i} (\log \pi_{\theta}(x,y) + i/\sigma^{2} \sum\nolimits_{i,j} s_{\theta}(i,j)\, f(x-i, y-j) \tag{4-9}$$

Where:

$f(x,y)\ and\ \acute{f}(x,y)$ are the original image and the output test statistic. $\propto (x,y)$ is the most probable line orientation at point (x,y). The stick with angular value of $\theta\ is\ represented\ as\ s_{\theta}(i,j)\ and\ W(x,y)$ is a mask function which has unit value on the union of the support of the sticks and zero-valued elsewhere.

$\pi_{\theta}(x, y)$ is the computed probability of a line at orientation θ at point $(x, y) and\ I(expr)$ is the indicator function equal to one if "expr" is true and zero if it is false.

Stick filter has two parameters that affect the sticks matrix size and the stick thickness. Figure 4-8 shows the effect of increasing the sticks matrix size on the filtered image. A thresholded image is added for each filtered image to help better show the border smoothing effect of this filter.

To obtain good results, the sticks should be made longer than the correlation length of the noise, yet shorter than the distance over which the boundaries appear to be straight lines. In our application, we have limited the stick matrix to 101 and stick thickness to 3.

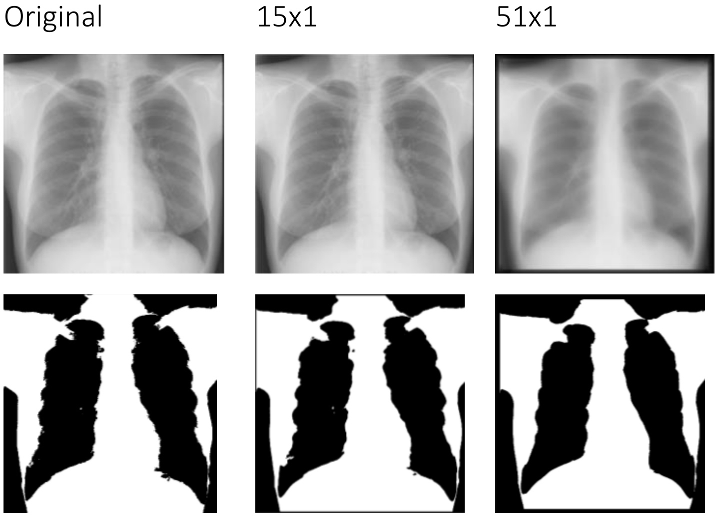

Original 15x1 51x1

Figure 4-8: Effect of increasing stick filter matrix size

4.1.3 Unsharp Masking Filter:

The unsharp masking filter [67,68] has the effect of making edges and fine detail in the image crisper. This technique comes from a publishing industry process, in which an image is sharpened by subtracting a blurred (unsharp) version of the image from itself to create the mask that is added to the original image. This process can be summarized as follows:

- Blur the original image
- Subtract the blurred image from the original which produces the mask
- Add the mask (produced in previous steps) to the original image.

This can be expressed as follows:

$$g(x,y) = f(x,y) + k * \ g_{mask}(x,y) \tag{4-10}$$

Where $f(x,y)$ is the original image and $g_{mask}(x,y)$ is the mask resulting from subtracting blurred version of the image from itself.

Original Image Sharpened Image

Figure -4-9: Effect of Unsharp Masking

In Figure *-4-9*, we can see a crisper image, and sharper border between black and the moon surface. Also, if we look at the moon surface, we can see crisper and sharper circles on the surface. Unsharp masking has no parameters to tweak.

4.1.4 Wiener:

Wiener filtering [69,70] or minimum mean square error minimizes the overall mean square of the uncorrupted image and its estimate. Wiener filter computes an estimate of an unknown signal by filtering an input signal that consists of the desired signal to estimate what has been corrupted by additive noise. The Wiener filter can be used to filter out the noise from the corrupted signal to provide an estimate of the underlying signal of interest. The Wiener filter is based on a statistical approach, and a more statistical account of the theory is given in the minimum mean square error (MMSE):

$$F^{\wedge}(u,v) = \left[\frac{H^{*}(u,v)\ S_f(u,v)}{S_f(u,v)|H(u,v)|^2 + \ S_\eta(u,v)}\right] G(u,v) \tag{4-11}$$

Where *H(u,v) and H*(u,v)* are the degradation function and its complex conjugate respectively. Power spectrum of the undegraded image is represented by $S_f(u,v)$ while $S_\eta(u,v)$ is the noise power spectrum

Original image Wiener [20 20] Wiener [30 30]

Figure 4-10: Applying Wiener filter with different window sizes

Wiener filter uses window that passes through the original image to apply the filter properly. Increasing this window beyond $[30,30]$ leads to a huge loss of lungs' border information. This window size is used as the maximum for wiener filter.

4.1.5 Average Filter:

This filter replaces processed pixel's grey level in image with the average of neighborhood pixels' grey level values. Average filter [73] gives a simple noise reduction method by reducing the intensity variation between pixels and neighboring pixels. This filter has two parameters that determine the window size to mask all pixels surrounding the processed pixel. The value of the restored pixel $f^{\wedge}(x, y)$ is the average value in the area defined by the neighborhood S_{xy} centered at the point (x, y). This can be expressed by the following formula:

$$f^{\wedge}(x, y) = \frac{1}{m * n} \sum_{(s,t) \in S_{xy}} g(s, t) \tag{4-12}$$

Figure 4-11 shows different sizes of averaging mask:

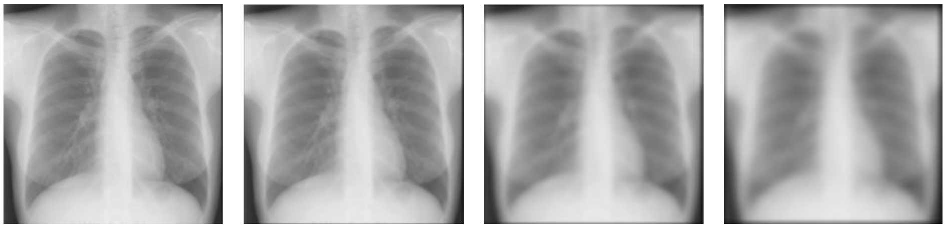
1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

Averaging [3 3]

1/25	1/25	1/25	1/25	1/25
1/25	1/25	1/25	1/25	1/25
1/25	1/25	1/25	1/25	1/25
1/25	1/25	1/25	1/25	1/25
1/25	1/25	1/25	1/25	1/25

Averaging [5 5]

Figure 4-11: Averaging Masks with different sizes



Original image Average [10 10] Average [30 30] Average [50 50]

Figure 4-12: Effect of averaging mask windows size when applied on Lung X-Ray image

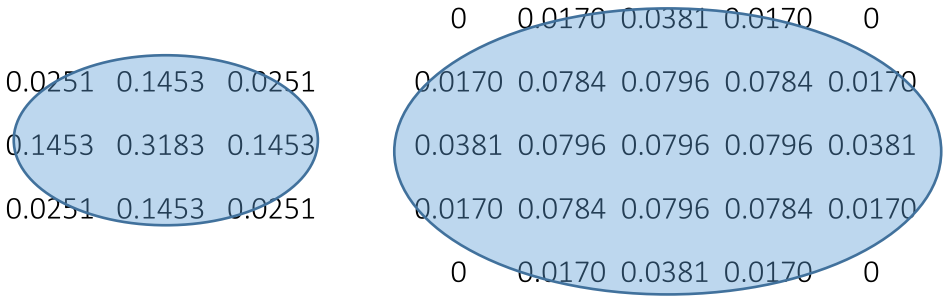
Figure 4-12 shows the effect of increasing the average window size. The last picture shows some loss of border information which might affect segmentation. Average filter in our optimizer is limited to 50x50 window size.

4.1.6 Disk Filter

Disk filter [73] performs similar calculations to the average filter, except that the average neighborhood has a disk shape instead of square in addition to weighted pixels such that it emphasizes the central pixel. The following equation (4-13) shows a general implementation of average filtering that is used in disk averaging as well as in different shapes

$$g(x,y) = \frac{\sum_{s=-a}^{a}\sum_{t=-b}^{b} w(s,t) f(x+s, y+t)}{\sum_{s=-a}^{a}\sum_{t=-b}^{b} w(s,t)} \qquad (4\text{-}14)$$

Figure 4-13 show two samples of disk shapes with different sizes used by our optimizer:



0.0251	0.1453	0.0251
0.1453	0.3183	0.1453
0.0251	0.1453	0.0251

Disk shape with 1-pixel radius

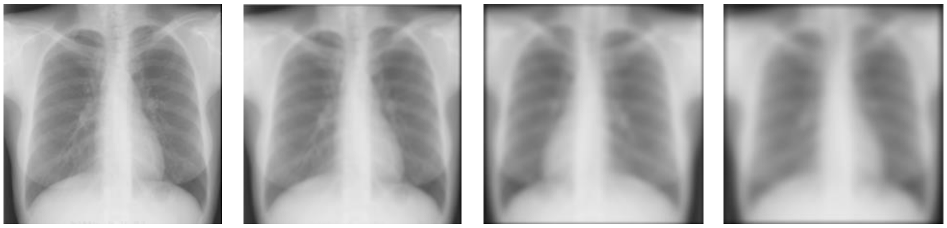
0	0.0170	0.0381	0.0170	0
0.0170	0.0784	0.0796	0.0784	0.0170
0.0381	0.0796	0.0796	0.0796	0.0381
0.0170	0.0784	0.0796	0.0784	0.0170
0	0.0170	0.0381	0.0170	0

Disk shape with 2 pixels' radius

Figure 4-13: Disk shape filter mask with different sizes

This filter offers noise reduction and smoothing ability tool for our optimizer. It has only one parameter, which is the disk radius.

Figure 4-14 shows that increasing disk radius to 30 would blur the image border and might affect later segmentation accuracy. The DE optimizer limit the maximum radius to 30 for our application



Original Image Disk 10 Disk 20 Disk 30

Figure 4-14: Effect of disk averaging filter with increasing radius size

4.1.7 Gaussian Filter:

This filter replaces processed pixel with Gaussian weighted average of the target pixel and surrounding neighborhood pixels [72,73]. Weights are designed to give higher importance to the central pixel. Other pixels are weighted inversely as a function of their distance from central pixel. Gaussian filter mask is generated using equation (4-14):

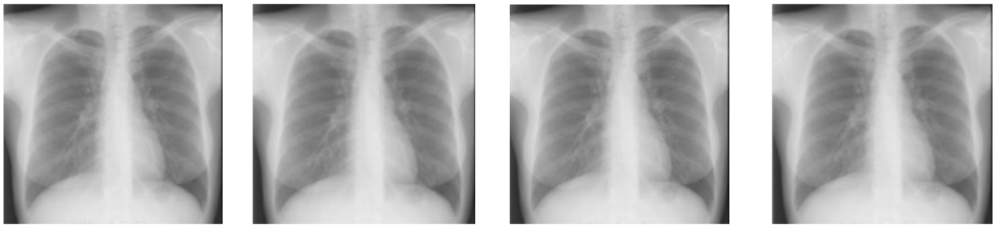
$$g(x,y) = \frac{1}{2\pi\sigma^2} * \ e^{-\frac{x^2+y^2}{2\,\sigma^2}} \tag{4-15}$$

Figure 4-15 shows two Gaussian filter mask samples generated by Matlab.

			0.0001	0.0020	0.0055	0.0020	0.0001
0.0439	0.1217	0.0439	0.0020	0.0422	0.1171	0.0422	0.0020
0.1217	0.3377	0.1217	0.0055	0.1171	0.3248	0.1171	0.0055
0.0439	0.1217	0.0439	0.0020	0.0422	0.1171	0.0422	0.0020
			0.0001	0.0020	0.0055	0.0020	0.0001

Figure 4-15: Two sizes of Gaussian filter masks (3x3) and (5x5), sigma 0.7

Gaussian filters provide a good smoothing with less blurring compared to other averaging filters mentioned earlier, which allows better preservation of the main target of this image processing chain.



Original Image Gaussian [50 50] Gaussian [100 100] Gaussian [200 200]

Figure 4-16: Effect of Gaussian filter with increasing windows size

Increasing Gaussian filter windows size while maintaining sigma = 3 in Figure 4-16 produces almost identical results which could be explained by looking at the filter surface produced by Matlab (Figure 4-17):

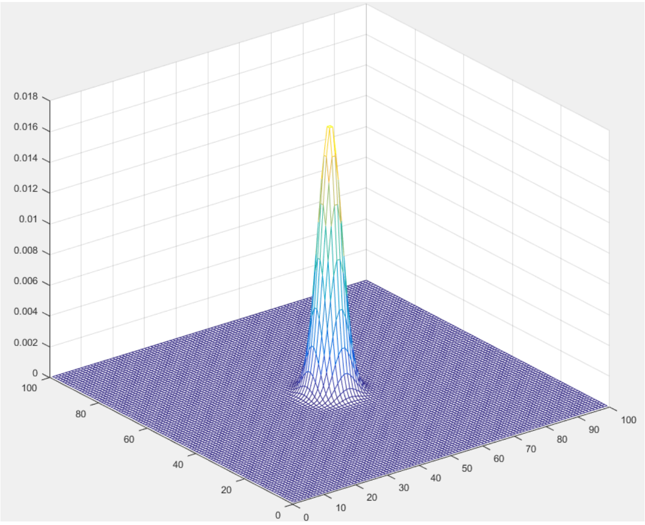


Figure 4-17: Gaussian filter graph with 100x100 window size, sigma = 3

Figure 4-17 shows Gaussian filter for window size $[100,100]$. It can be easily noticed that the filter takes in consideration only the central pixels around the target pixel window to be weight averaged. It means that increasing window size while maintaining sigma has no effect after a certain window size.

On the other hand, increasing sigma using the same window size increases the number of pixels that are weight averaged in the Gaussian window. Gaussian filter window size should be chosen such that most of the Gaussian bell is included in the window. Choosing window size, which is five times the sigma, value produces Gaussian filter such as the filter value at the corner is zero and other edge values are negligible.

To reduce the dimensionality of the problem, we suggest removing dimension variables that are related to Gaussian window size from the chromosome and restrict the variables needed for this filter to only sigma value and calculate the Gaussian filter mask based on size = 5xSigma.

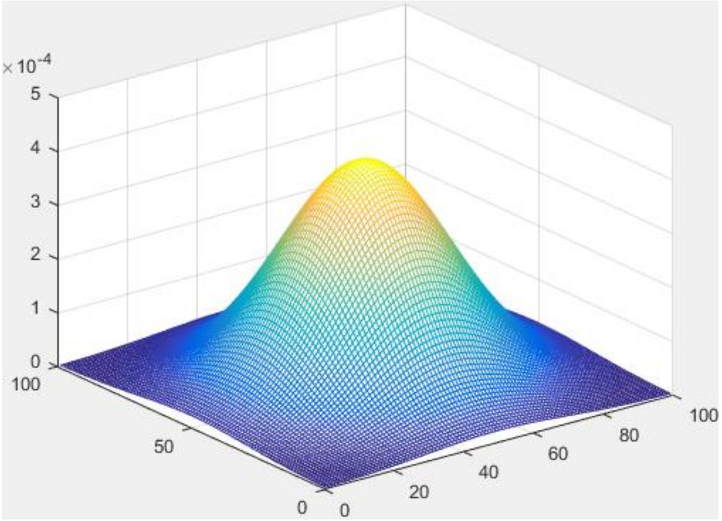
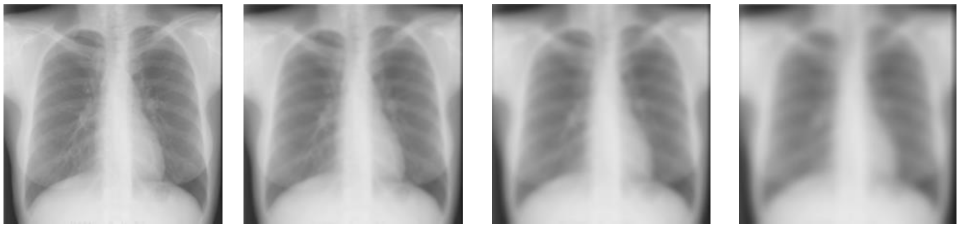


Figure 44-18: Gaussian bell windows' size = 100, sigma = 20

Figure 4-19 shows the effect of increasing sigma. When sigma is 50, the border data tend to be lost. Based on this experiment, sigma will be limited to 50.



Original Image Sigma 15 Sigma 30 Sigma 50

Figure 4-19: Effect of Gaussian filter with increasing sigma

4.1.8 Laplacian of Gaussian (LoG) Filter:

This filter [74,75,76] is designed based on two principles set by Marr and Hildreth. First, image intensity changes are not independent of image scale, which requires using operators of different sizes (i.e., tuned operators); and second, that pixel at edges have peak change in intensity for first derivatives and zero crossing for a second derivative.

Marr and Hildreth used the laplacian operator ∇^2 to develop LoG filter as this filter fulfills the above mentioned conditions. This ∇^2operator for 2-D Gaussian function is given as below:

$$G(x,y) = e^{-\frac{x^2+y^2}{2\,\sigma^2}} \tag{4-16}$$

Using the Laplacian operator on the above Gaussian equation yields to the Laplacian of Gaussian formula used in these types of filters:

$$\nabla^2 G(x,y) = \left[\frac{x^2 + y^2 - 2\sigma^2}{\sigma^4}\right] e^{-\frac{x^2+y^2}{2\,\sigma^2}} \tag{4-17}$$

The ability to tune the operator to target any desired scale and the fact that it is a derivative operator capable of detected changes in intensity and identifying edges makes this operator a very good tool for our chain optimizer.

LoG filter has three parameters; two parameters are related to the determination of the mask size and the third parameter determine the Gaussian sigma. Figure 4-20 shows the effect of different parameters of this filter on a Lung X-Ray image:

Original image sigma=0.1 sigma=0.4 sigma=1

Figure 4-20: Effect of LoG filter with fixed window size [20 400] and increasing sigma value

Even though the last image shows black, it actually contains data that can be seen if we apply Otsu thresholding. Figure 4-21 shows the thresholding result.

Figure 4-21 Showing pixel information contained in black lung X-Ray image filtered with LoG with sigma = 1

Based on experiments, we can determine the maximums for this filter as follows:

- Window size maximum [50% 50%] of the image size
- Sigma maximum 3

4.1.9 Laplacian Filter:

Laplacian operator [79] [80] is the second derivative defined as follows for two dimensions:

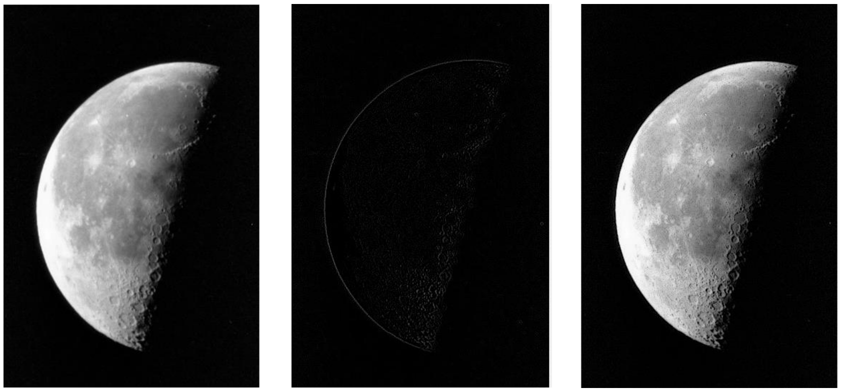
$$\nabla^2 = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \tag{4-18}$$

Second derivatives produce spike values when applied to rapidly changing function sections. This make this operator very important to highlight intensity discontinuities in an image and deemphasizes regions with slowly varying intensity levels. Applying this filter to an image highlight edge lines on a dark background, which can then be added to the original image. This method produces a sharpened image. Applying this filter on digital images requires using a discrete version of Laplacian, which is expressed as follows:

$$\begin{aligned}\nabla^2 f(x,y) = f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) \\ - 4f(x,y)\end{aligned} \tag{4-19}$$

This can be expressed in matrix masks as one of the following

$$\begin{bmatrix}0 & 1 & 0\\ 1 & -4 & 1\\ 0 & 1 & 0\end{bmatrix}\begin{bmatrix}1 & 1 & 1\\ 1 & -8 & 1\\ 1 & 1 & 1\end{bmatrix}\begin{bmatrix}0 & -1 & 0\\ -1 & 4 & -1\\ 0 & -1 & 0\end{bmatrix}\begin{bmatrix}-1 & -1 & -1\\ -1 & 8 & -1\\ -1 & -1 & -1\end{bmatrix}$$



Original Image

Filtered with Laplacian

Original minus Laplacian

Figure 4-22: Effect of Laplacian filter applied on moon image

This filter has one parameter that controls generating the above matrices. The range is fixed between 0 and 1 by Matlab Laplacian implementation. Figure 4-22 shows the effect of implementing this filter.

Applying Laplacian produces a dark image with greyish borders (middle image). To emphasize the border and enhance it, we can subtract/add this Laplacian filtered image from/to the original image (right side image). This produces a sharpened and crisper image, as you can see in Figure 4-22.

4.1.10 Prewitt Filter:

Prewitt filter [81] [82] calculates the gradient of the image intensity at each point. This identifies the direction of the largest possible increase in the pixel intensity from light to dark. Having this filter in pre-processing filters gives the chain optimizer an essential tool to enhance edges for later processing stages in the change.

It is considered a first derivative operator that calculates the gradient in both directions x and y. The gradient of image $f(x,y)$ at location (x,y) is defined as the vector:

$$\nabla f = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} \tag{4-20}$$

Prewitt operator uses 3x3 mask size to calculate the gradient in both direction x and y. An approach using it is developed based on the equations [4-17] [4-18]

$$G_x = (z_7 + z_8 + z_9) - (z_1 + z_2 + z_3) \tag{4-21}$$

$$G_y = (z_3 + z_6 + z_9) - (z_1 + z_4 + z_7) \tag{4-22}$$

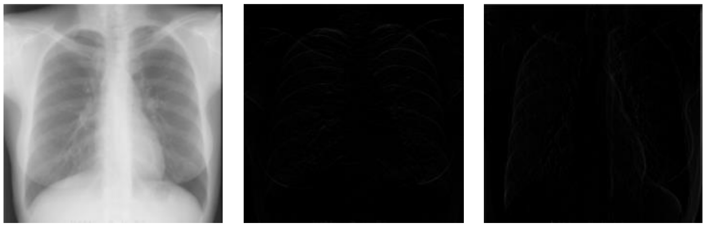
Which calculates the difference between first and third column in the 3x3 image mask. Both equations can be expressed in masks which are shown in Figure 4-23. There are no parameters for Prewitt filter. We can apply it in the vertical or horizontal direction.

$$\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \qquad \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

$$G_x \qquad\qquad\qquad G_y$$

Figure 4-23: Gx,y Masks representing [4-9,10] equation as matrices

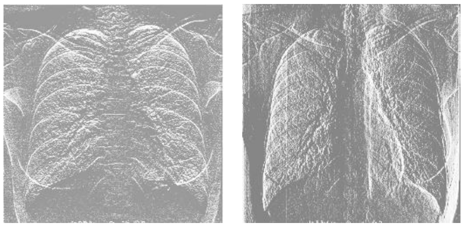
Figure 4-24 shows the effect of applying Prewitt on lung image. The first row shows almost black images as a result of applying Prewitt. We can barely see the edges. Second line shows the histogram equalization to manifest the effect of Prewitt operator and how it emphasizes the horizontal or vertical edges. The results of Prewitt filter are normally added to the image to enhance the edge on the original image (second row in the Figure 4-24). This prepares it for further processing.



Original Image Horz. Prewitt Vert. Prewitt

Original Image Horz. Prewitt Vert. Prewitt

Figure 4-24: First row show Prewitt output when applied on Lung X-Ray image (black images), second row of images is the result of adding Prewitt output to original image to emphasis edges



(a) Horz. Prewitt (b) Vert. Prewitt

Figure 4-25: Applying histogram equalization on Prewitt filter

Figure 4-25 shows the result of applying histogram equalization on Prewitt filter black image output to better illustrate its effect in emphasizing horizontal edges in Figure 4-25 (a) and vertical edges in Figure 4-25 (b).

4.1.11 Sobel

Sobel [79,81,82] is a gradient operator similar to Prewitt except that it emphasizes its mask on the center of rows and columns. Sobel provides the optimizer with another technique to enhance image edges. Sobel operator uses 3x3 mask size to calculate the gradient in both direction x and y. An approach using it was developed based on the equations (4-19) (4-20)

$$G_x = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3) \tag{4-23}$$

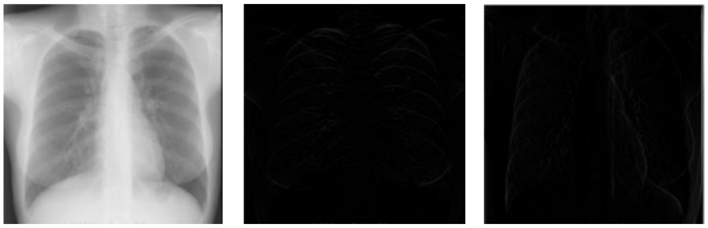
Which G_x calculates the difference between the first and third row in the 3x3 image mask

$$G_y = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7) \tag{4-24}$$

Which G_y calculates the difference between the first and third column in the 3x3 image mask. In both equations (4-19) (4-20), Sobel operator multiplies the central cell by two in both rows and columns to emphasize the central pixel in filtered images. Based on the above mentioned equations, we can create the following Sobel masks:

$$\underset{G_x}{\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}} \qquad \underset{G_y}{\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}}$$

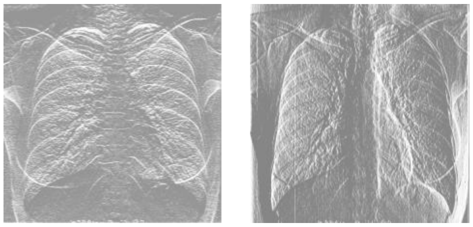
There are no parameters for Sobel filter. We can apply it in the vertical or horizontal direction. Figure 4-26 shows the effect of applying Sobel on the lung image.



Original Image Horizontal Sobel Vertical Sobel

Figure 4-26: Effect of applying Sobel filter on Lung X-Ray image

The first row in Figure 4-26 shows almost black images as a result of applying Sobel. These black images actually have the data of edges that could be used to enhance edges if it was added to the original image as seen in the second row in Figure 4-26. This prepares it for further processing. Second line shows histogram equalization to manifest the effect of Sobel operator and how it emphasizes the horizontal or vertical edges. To verify the true effect, we apply histogram equalization as shown in Figure 4-25.



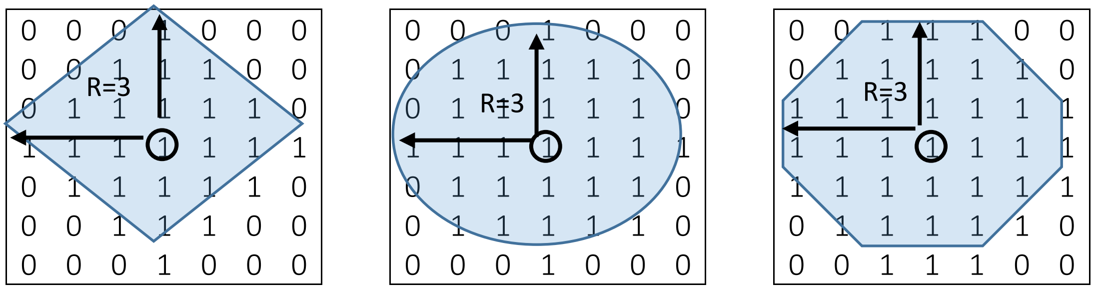
(a) Horz.Sobel (b)Vert. Sobel

Figure 4-27: Histogram Equalization applied on sobel filtered images

Figure 4-25 (a) clearly shows horizontal white lines clearly visible compared to vertical lines which is an indication of horizontal edge emphasis by Sobel, while (b) shows vertical white lines are clearly visible compared to horizontal lines which indicates a vertical edge emphasis by Sobel.

4.1.12 Morphological filters:

Morphological filters [85] focus on extracting features, attributes, and meaning from images. They are used as probes to examine an image for specific attributes. The core function of these filters depends on using structural elements that passes through every pixel of the image similar to masks in spatial filters and convolution operation. Figure 4-28 shows different shapes and sizes that are used as structural elements. More structure elements are shown in Figure 4-29: **Different shapes of structural elements**. In sequence: first SE has an arbitrary shape determined by the location of ones, second SE shows a line of seven pixels with zero degree which can be changed to any angle and finally, right most SE is the pair SE which contains two elements: One element is located at the origin, and the other at an offset specified by 2D parameter.



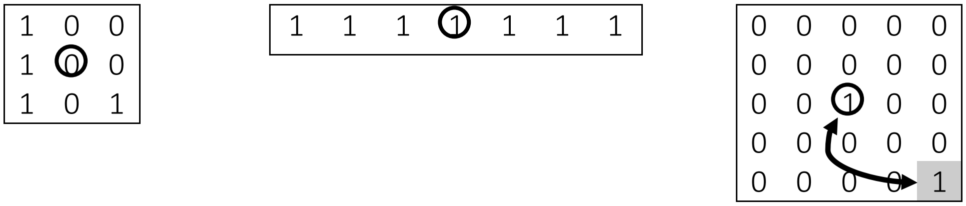
Diamond Structural Element

Disk Structural Element

Octagon Structural Element

Figure 4-28: (a) Diamond (b) Disk (C) Octagon Structural Elements

Different flat shapes are shown in Figure 4-28 with radius R=3. These structure elements have one parameter:



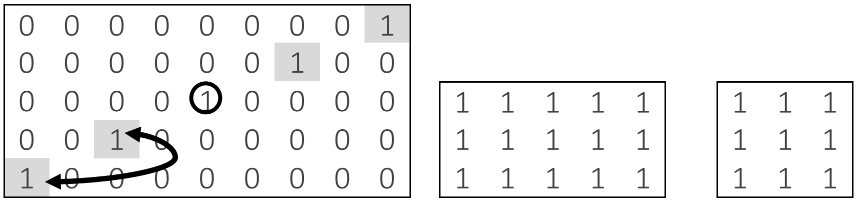
Arbitrary structural Element

Line Structural Element=7, Deg=0

Pair Structural Element (offset (2,2))

Figure 4-29: Different shapes of structural elements

Figure 4-30 shows the periodic line SE that accepts two parameters P to determine the number of members in this SE using the formula $2p + 1$. The other parameters for the periodic line SE is a vector which determines the offset "V" of other members in the SE. The first member is located at the origin (center of the SE) and other members are located at 1*V, -1*V, 2*V, -2*V…., P*V, -P*V.



Periodic line SE v=[1 -2]

Rectangle SE

Square SE

Figure 4-30: Different shapes of structural elements

The structural elements explained above are used by the four morphological filters explained in the following four sections which used our image processing chain. The following sections demonstrate the effect of these filters on sample images and also develop the maximum and minimum vectors of their parameters based on testing different values of parameters.

4.1.12.1 Erosion

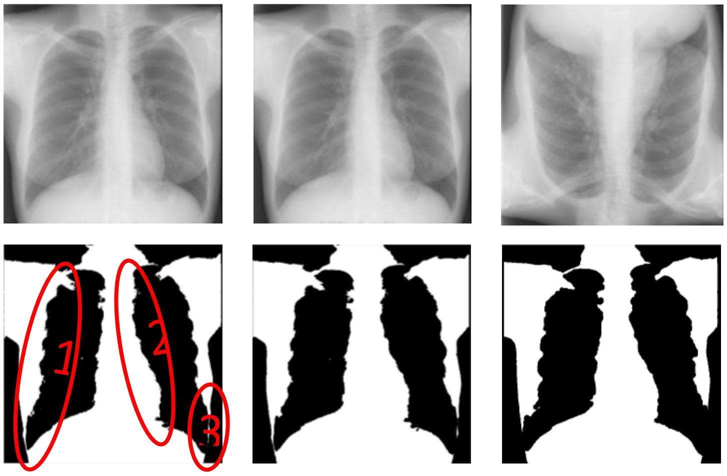
The erosion [86] [87] of a grayscale image $f(x, y)$ by a flat structuring element “b” at any location (x, y) is the minimum value of the intensity level in the region specified by the structuring element “b”, when the origin of “b” is located at the point (x, y) of the image. This is represented by the following mathematic equation:

$$[f \ominus b](x, y) = min_{(s,t)\in b}\{f(x + s, y + t)\} \tag{4-25}$$

Where s and t are local variables used to scan pixels within the boundaries of the structuring element “b”. x and y are incremented such that it visits all pixels in the image. Erosion accepts one parameter, the structure element, which in turn accepts between two to three parameters. For demonstration purposes, we use a disk shaped SE with diameter of three and eleven pixels. Effect of erosion is better demonstrated by thresholding both original and eroded images in the Figure 4-31. The regions 1 and 2 (magnified in Figure 4-32) show clearly that the original image has roughly enveloped contour for the lungs while in eroded image, it can be seen as a smooth envelope. On the other hand, region 3 in the original image is almost disconnected, while the eroded image has more connected points to the frame of the image. This is due to the way of erosion works in grayscale image; it chooses the lower intensity in the SE windows, which yields in increasing dark areas.

Increasing SE diameter makes the envelope smoother, but it fills the gaps and causes segmentation problems as the frame is considered part of the lung (see Figure 4-31) in the eroded image using disk radius of six pixels. Based on this, we limit SE diameter size to a maximum of 2% of image size, which gives enough range for the DE optimizer to maneuver and find best SE size for erosion filter.

Figure 4-32 shows zone one in Figure 4-31, magnified to better illustrate the smoothing effect of erosion.



Original image

Eroded disk shape SE, Radius=3

Eroded disk shape SE, Radius=6

Figure 4-31: Effect of Erosion applied on Lung X-Ray image

Original Image

Eroded Image

Figure 4-32: Magnification of zone 1 in fig. 4-31 showing the effect of erosion

4.1.12.2 Dilation

Similar to erosion filter the dilation [85,86] by a flat SE "b" at any location (x, y) is defined as the maximum value of the image in the window specified by $b^\wedge$ when the origin of $b^\wedge$ is at the location (x, y). The mathematical formula of the above mentioned definition is as follows:

$$[f \oplus b](x, y) = max_{(s,t)\in b}\{f(x-s, y-t)\} \qquad (4\text{-}26)$$

Where s and t are local variables used to scan pixels within the boundaries of the structuring element "b". x and y are incremented such that it visits all pixels in the image. Dilation parameters are the same as in erosion. Figure 4-33 demonstrates the effect of dilation using a disk shaped SE with diameter of three and eleven pixels; respectively.

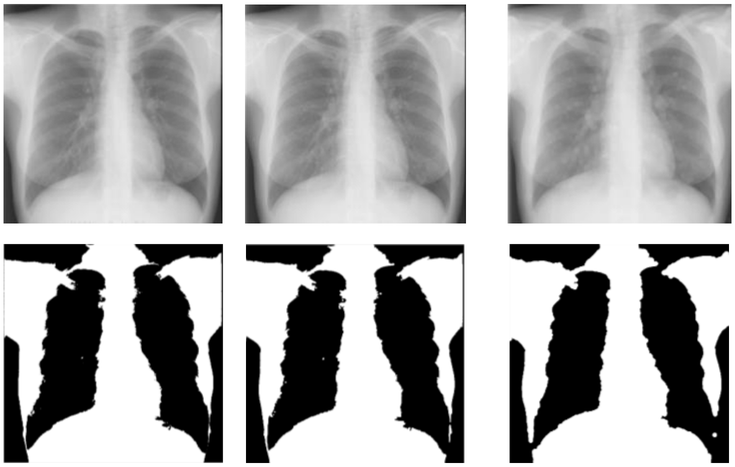


Figure 4-33: Effect of Dilation with three-pixel diameter disk shape SE

Similar to Erosion effect, lung envelops are smoother, but they have different sizes and shapes of the eroded image. Increasing structure of element size causes the similar effect like erosion. Lung envelope is smoother, but it has connections to the image frame which affects segmentation results.

Based on the results shown in this section, we can restrict the maximum values of the SE diameter to 2% of the image size, which give the DE optimizer sufficient range to get the best segmentation results.

4.1.12.3 Morphological Opening:

Morphological opening grayscale image [89] “f” by structure element “b” is achieved by eroding the image with SE “b”, then dilating the result with the same SE “b”. Mathematically it is expressed as follows:

$$f \circ b = (f \ominus b) \oplus b \tag{4-27}$$

Similar to other morphological filters, this filter accepts a structuring element which is determined by shape and diameter. Figure 4-34 shows the effect of this filter on a sample grayscale image using diameters of three and eleven pixels.

Increasing SE diameter causes the dark area to increase and the targeted feature to connect with the frame causing segmentation errors later. Restricting the SE diameter to 2% give enough diameter value range to DE to find the optimal value.

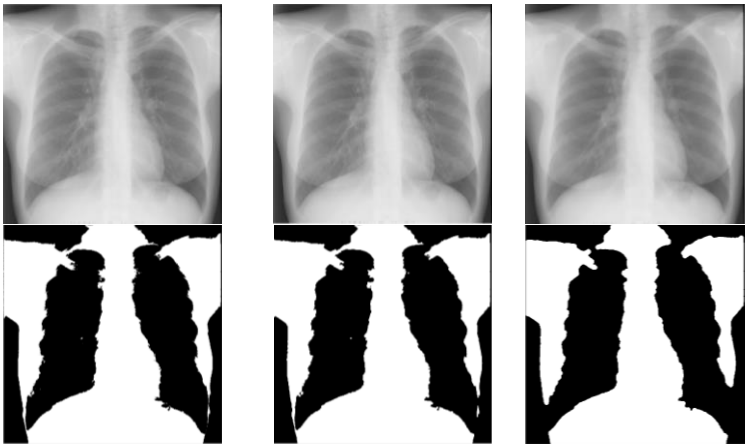

Original Image 3pixels SE disk 11 pixels SE disk

Figure 4-34: Effect of opening with disk shape SE

4.1.12.4 Morphological Closing:

Morphological closing grayscale image [89] [90] “f” by structure element “b” is achieved by dilating the image with SE “b”, then eroding the result with the same SE “b”. Mathematically, it is expressed as follows:

$$f \cdot b = (f \oplus b) \ominus b \tag{4-28}$$

Similar to other morphological filters, this filter accepts a structuring element which is determined by shape and diameter. Figure 4-35 shows the effect of this filter on a sample grayscale image using three and eleven-pixel diameter, respectively. Similar to other morphological filters, increasing SE diameter causes the dark area to increase and the targeted feature to connect with the frame causing segmentation errors later. Restricting the SE diameter to 2% gives enough diameter value range to DE to find the optimal value.

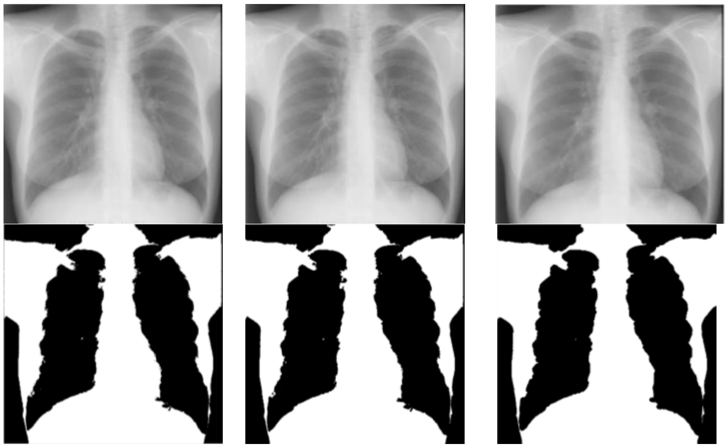

Original Image 3 pixels SE disk 11 pixels SE disk

Figure 4-35: Effect of closing with disk shape SE

4.2 Thresholding algorithms

Built around splitting image into classes based on intensity levels. Thresholding algorithms form a very essential step in the image segmentation; they are simple and effective with low computation requirements. In the following chapters, we review mathematical principles of thresholding algorithms used in our image processing chain.

4.2.1 Otsu

The algorithm [20] [21] assumes that the image contains two classes of pixels following bi-modal histogram (foreground pixels and background pixels), then calculates the optimum threshold separating the two classes so that their combined spread (intra-class variance) is minimal, or equivalently (because the sum of pairwise squared distances is constant), so that their inter-class variance is maximal. This algorithm has one parameter, the threshold level, which is used to split the image into two classes. This parameter has intensity level range which is 0 to 255. Otsu algorithm target is maximizing the between-class variance which is defined as follows:

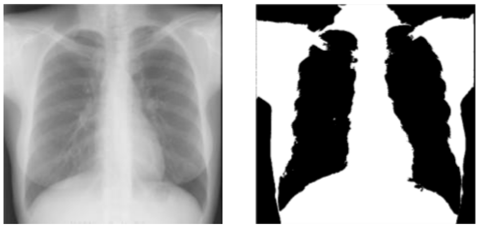
$$\sigma_B^2 = P_1 P_2 (m1 - m2)^2 \tag{4-29}$$

Where m1 and m2 are the means of the two separated classes as a result of thresholding image using k value. P1 and P2 are probabilities for each class resulting from the thresholding with k. It is calculated as follows:

$$P_1(k) = \sum_{i=0}^{k} P_i \tag{4-30}$$

The between-class variance calculated by the equation shows that the larger difference between m1 and m2, the larger between-class variance, which indicates that "σ_B^2" is a measure of separability. The target for Otsu algorithm is maximizing σ_B^2.

Otsu calculates $\sigma_B^2(k)$ for every possible thresholding value for the image and chooses the k value that maximize σ_B^2. This algorithm doesn't have any parameter. Figure 4-36 shows results of using it on a sample image.



Original Image Thresholded Image

Figure 4-36: Effect of Otsu Thresholding

4.2.2 Kittler (Minimum Error Method)

This algorithm [91,92,93] uses arbitrary grey level to threshold the image, and models each of the two resulting pixel populations by probability density function using the histogram of each class as an estimate PDF p(g). This algorithm assumes that each of these two components or class “i” p(g/i) are normally distributed with mean μ_i standard deviation σ_i, and probability P_i:

$$P(g) = \sum_{i=1}^{2} P_i p(g/i) \tag{4-31}$$

$p(g/i)$ is given by:

$$p(g/i) = \frac{1}{\sigma_i \sqrt[2]{2\pi}} \, exp(-\frac{(g-\mu_i)}{2\sigma_i^2}) \tag{4-32}$$

Using both equations (4-26) and (4-27) then applying the logarithm on both sides of the resulting equation and rearranging, we get the following quadratic equation (4-28):

$$\frac{(g-\mu_1)^2}{\sigma_1^2} + log\,\sigma_1 - 2\,log\,P_1 = \frac{(g-\mu_2)^2}{\sigma_2^2} + log\,\sigma_2 - 2\,log\,P_2 \tag{4-33}$$

This equation can be solved with difficulties due to the fact that $\sigma, \mu, and\ P$ are not known, even though they can be estimated. Kittler and Illingworth [91] have suggested following function to be minimized for the thresholding variable “t”:

$$\begin{aligned} J(t) = 1 + 2(P_1(t).\,log\,\sigma_1(t) + P_2(t).\,log\,\sigma_2(t)) \\ - 2(P_1(t).\,log\,P_1(t) + P_2(t).\,log\,P_2(t)) \end{aligned} \tag{4-34}$$

Where

$$P_1(t) = \sum_{g=0}^{t} h(g) \qquad P_2(t) = \sum_{g=t+1}^{255} h(g)$$

$$\mu_1(t) = \frac{\sum_{g=0}^{t} g.h(g)}{P_1(t)} \qquad \mu_2(t) = \frac{\sum_{g=t+1}^{255} g.h(g)}{P_2(t)}$$

$$\sigma_1^2(t) = \frac{\sum_{g=0}^{t} h(g).(g - \mu_1(t))^2}{P_1(t)}$$

$$\sigma_2^2(t) = \frac{\sum_{g=0}^{t} h(g).(g - \mu_2(t))^2}{P_2(t)}$$

Thresholding value “t” that minimizes the function j(t) in the equation (4-29) is the optimum thresholding value.

Original Image Thresholded Image

Figure 4-37: Kittler Thresholding

Figure 4-37 shows that optimal thresholding using Kittler algorithms gives white space. Proper pre-thresholding processing is required to get the best out of this filter, which can be achieved by the optimized image processing chain.

4.2.3 Minimum cross-entropy threshold selection

MCE [94] is a non-metric measure between a posteriori probability distribution q(x) and a priori distribution p(x). This measure can be used to determine the optimum threshold.

$$H_{CE}(q,p) = \sum_{x=1}^{N} q_x \, . \log \frac{q_x}{p_x} \tag{4-35}$$

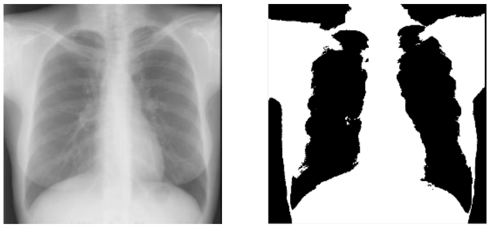
Where:

$$\sum_{x=1}^{N} p_x = \sum_{x=1}^{N} q_x \quad (= 1) \tag{4-36}$$

For the purposes of threshold selection, the original grey-scale image is considered as the prior distribution. Assuming the image is thresholded at the level “T”, we can develop the above and below threshold level “T” mean functions $\mu_0(T)$ and $\mu_1(T)$, which can be used to represent the binary image. Compensating the means in equation [4-30] and splitting the sum to above and below threshold “T” produce the following equation:

$$H_{CE}(T) = \sum_{g=a}^{T} f_g.\mu_0(T).\log\frac{\mu_0(T)}{g} + \sum_{g=T+1}^{b} f_g.\mu_1(T).\log\frac{\mu_1(T)}{g} \tag{4-37}$$

Where, f_g is the number of pixels having grey-level g while a and b represent the minimum and maximum grey level in the image.



Original Image Thresholded Image

Figure 4-38: Minimum Cross-Entropy thresholding

Figure 4-38 above shows MCE thresholding result when applied to sample Lung X-ray image.

4.2.4 Triangle method:

Zack et al. [95] in 1977 introduced the algorithm as demonstrated in Figure 4-39 is based on constructing a line between the maximum value of a normalized histogram and the lowest value, which allows locating a point like “A”. The optimal threshold is then chosen at a fixed offset of this point.

The distance L between the line and the histogram using normal line is computed for all values from a to b. The level, where the distance between the histogram and the line is maximal, is the threshold value (level). This technique is particularly effective when the object pixels produce a weak peak in the histogram.

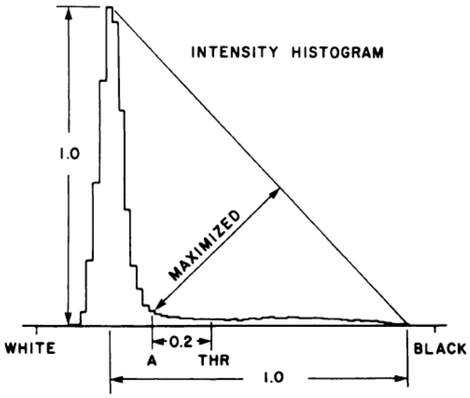
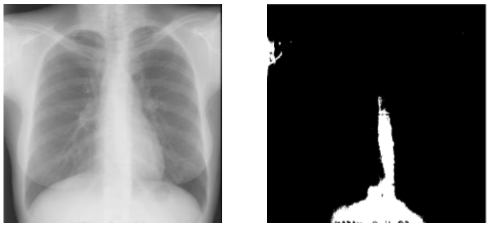


Figure 4-39: Triangle thresholding technique

Figure (4-39) shows thresholding results on a sample image by using this algorithm.



Original image   Thresholded image

Figure 4-40: Triangle thresholding

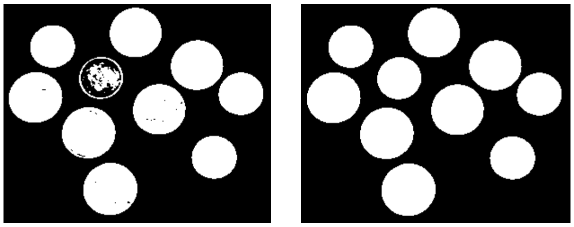
This thresholding technique has no parameters to be optimized. The results on a non-processed image as shown in Figure 4-40 is very poor. These results can be improved if we apply pre-processing filters and contrast enhancers, which is what the optimized IPC will do if this thresholding technique was chosen as part of the IPC.

4.3 Post Processing Filters

The optimizer uses these filters after binarizing the image using one of the thresholding techniques. These filters deal with binary images and the target is to fix any possible missing classified pixels using different types of techniques that are discussed in the following sections.

4.3.1 Hole Filling

Soille [96] [97] is defined a hole in a binary image as a set of pixels that corresponds to the regional minima, which are not connected to the image border. In filling hole algorithms, these pixels are filled with maximum value which is one in binary images. Figure 4-41 shows the effect of hole filling on a sample binary image.



Original Image Hole filled image

Figure 4-41: Hole Filling effect

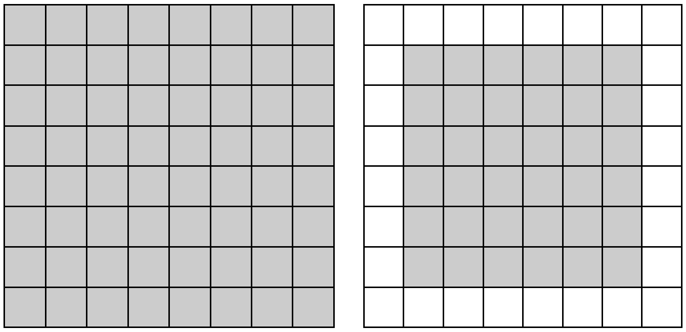
Hole filling used in our IPC has no parameter to be tuned by DE optimizer.

4.3.2 Erosion

Similar to morphological erosion for greyscale image, binary erosion [86] [98] uses the same structuring elements as explained earlier in Figure 4-28, Figure 4-29, and Figure 4-30. Erosion in post processing deals with binary images, and so the calculation formula of the erosion is different from the greyscale formula.

$$A \ominus B = \{z|(B)_z \sqsubseteq A\} \tag{4-38}$$

Which basically means that the structural element B origin visits every point z in the image A. The erosion of $A$by B is the set of points z, where the SE B is totally contained in A



Original Image Eroded Image

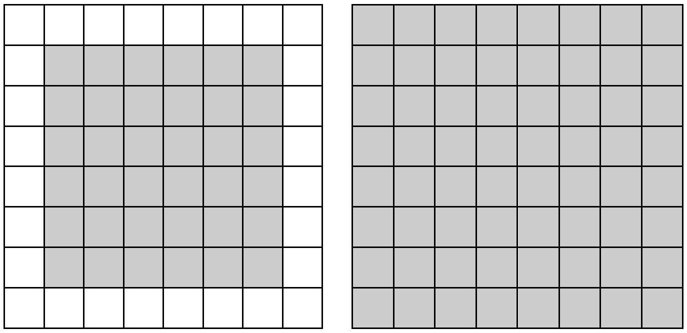
Figure 4-42: Effect of eroding with 3x3 structure element

Figure 4-42 shows how borders are eroded using Erosion algorithm. Erosion in our IPC has one parameter to be tuned, which is the SE diameter.

4.3.3 Dilation

Similar to morphological dilation for greyscale image, binary dilation [88] [98] uses the same structuring elements as explained earlier in Figure 4-28, Figure 4-29, and Figure 4-30. Dilation in post processing deals with binary images so the following formula is used to calculate dilation which is different from greyscale formula.

$$A \oplus B = \left\{z | (\widehat{B})_z \cap A \neq \phi\right\} \tag{4-39}$$



Original Image Dilated by 3x3 element

Figure 4-43: Effect of dilation by 3x3 structure element

Equation (4-38) shows that dilating A by B results in the set of points z where A and reflection of B overlaps with at least one element. The effect of dilation in this case is the opposite of erosion, as we can see in Figure 4-43. Dilation algorithm used in our IPC has only one parameter, which is the diameter of the structuring element.

4.3.4 Opening

Opening [89] [99] binary image f by structure element b t is achieved by eroding the image with SE b then dilating the result with the same SE b

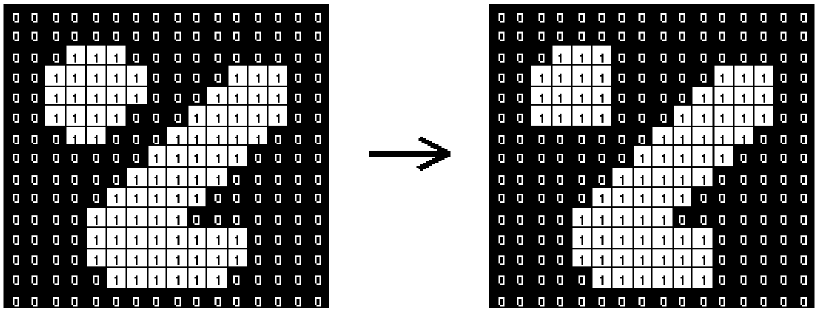
Mathematically, it is expressed as follows:

$$f \circ b = (f \ominus b) \oplus b \tag{4-40}$$

We can convert the above equation using set theory terminology as follows:

$$A \circ B = \bigcup \{(B)_z | (B)_z \sqsubseteq A\} \tag{4-41}$$

Which means that the opening image A by structuring element B is the union of all translates of B that fit into A. Similar to other morphological filters. This filter accepts a structuring elements explained earlier in Figure 4-28, Figure 4-29, and Figure 4-30 in section 4.1.12, which are determined by shape and diameter. Figure 4-44 shows the effect of this filter on a sample binary image.



Original Image

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	1	0	0	0	0	1	1	1	0	0
0	0	1	1	1	1	1	0	0	0	1	1	1	1	0	0
0	0	1	1	1	1	0	0	0	1	1	1	1	1	0	0
0	0	0	1	1	0	0	0	1	1	1	1	1	0	0	0
0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0
0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0
0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0
0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Opened Image

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	0	0	0	0	0	1	1	1	0	0
0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0
0	0	1	1	1	1	0	0	0	1	1	1	1	1	0	0
0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0
0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0
0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0
0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 4-44: Effect of applying opening

Figure 4-45 shows another sample of opening using circle structuring element and square image, applying the morphological opening equation [4-42], we can see that rolling the circle on the inside border of the square image and unionizing all the scanned points, we get a rounded square, which is the result of the morphological opening.

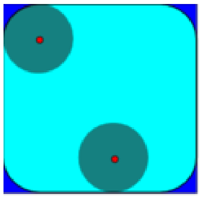


Figure 4-45: Morphological opening illustrated as rolling ball on the inside border of image

4.3.5 Closing

Closing [89] [99] binary image “f” by structure element “b” is achieved by dilating the image with SE “b”, then eroding the result with the same SE “b”.

Mathematically, it is expressed as follows:

$$f \cdot b = (f \oplus b) \ominus b \tag{4-43}$$

We can convert the above equation using set theory terminology as follows:

$$A \cdot B = \bigcup \{(B)_z | (B)_z \cap A \neq \phi\} \tag{4-44}$$

Interpretation of this equation (4-39) is similar to opening, except that the ball is rolling on the outside surface, as it can be seen in Figure 4-46.

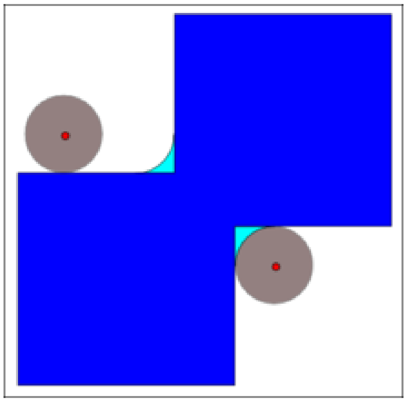


Figure 4-46: Morphological closing illustrated as rolling ball on the outside surface of image

The effect as illustrated in Figure 4-46, is rounding sharp corners on the outside face of the shape. Similar to other morphological filters, this filter accepts a structuring element which is determined by shape and diameter as illustrated in Figure 4-28, Figure 4-29, and Figure 4-30 in the section 4.1.12.

4.4 Contrast Enhancers

In most cases, pixels related to targeted features or borders reside in low contrast regions. Increasing image contrast causes a better segmentation of these features. Enhancer works in the spatial domain, where every pixel intensity level is replaced by new value based on a function, such that:

$$s = T(r) \tag{4-45}$$

Where s indicates the new values at point (x, y) and r is the old value, while T represents the transformation function. DE algorithm can choose up to three enhancers from a pool of eight enhancers to be integrated in the Image Processing Chain.

4.4.1 Contrast stretching (Global):

The contrast adjuster [100] is used in our IPC to map the intensity values in grayscale image “A” to new values in image “B”, such that data is saturated below set low value and above set high value of intensity values of the original image.

The algorithm uses either linear transformation function or gamma transformation function to map original image intensities to enhanced image contrast. Figure 4-47 shows possible transformation functions based on gamma values.

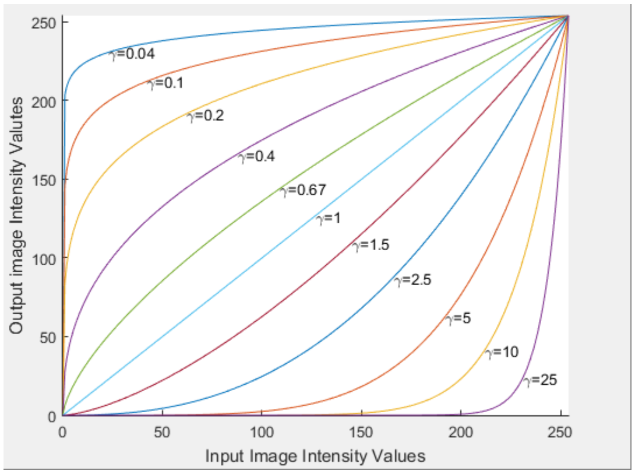
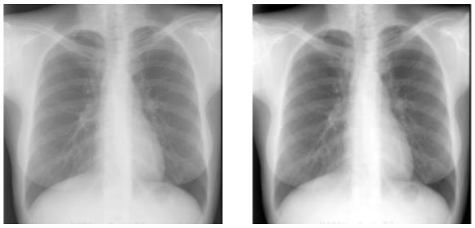


Figure 4-47: Gamma transformation function with different gamma values

Figure 4-48 shows the effect of applying this contrast enhancer on the sample image.



Original image   Stretched Contrast

Figure 4-48: Effect of contrast stretching

This stretcher accepts five parameters which determines current image intensity range to stretched to new range of gray levels. It also uses a fifth parameter to determine whether it uses gamma transformation function or keeps the transformation function linear. These parameters have a min-max range of [0-1]. Fifth parameter has one of two values of 0 or 1.

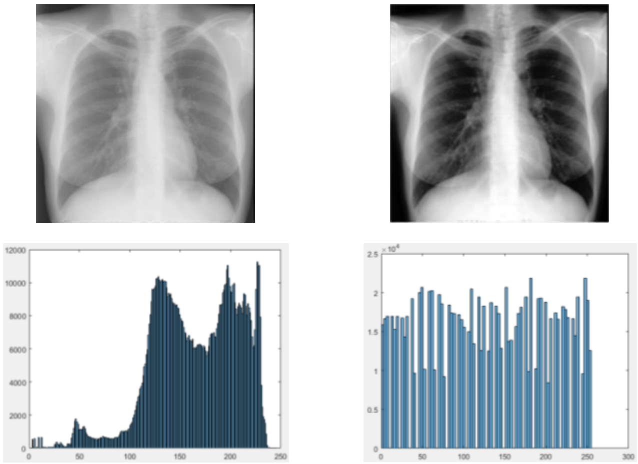
4.4.2 Histogram Equalization

Histogram equalizer [101] [102] transforms the intensity of input image “f”, so that the histogram of the output intensity image “J” histogram bins approximately matches the flat histogram with evenly gray level distribution among all possible gray levels that exist in the image. The transformation function of this contrast enhancer can be expressed by using information extracted from original image intensity level distribution as follows:

$$s_k = \frac{(L-1)}{MN}\sum_{j=0}^{k} n_j \qquad (4\text{-}46)$$

Where, MN is the image pixel count. s_k and $n_j$are the new calculated intensity level and number of pixels that have intensity level equal to k respectively. L: is the intensity levels existing in original image before transformation.

Figure 4-49 below shows the effect of histogram equalization.



Original Image Histogram Equalized Image

Figure 4-49: Illustration of histogram equalization effect on image histogram

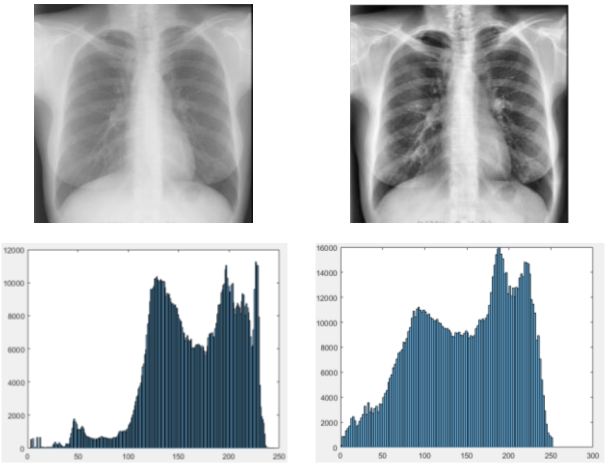
Histogram equalization algorithm used in IPC accepts only one parameter that determines the number of desired gray levels in the output image. The maximum number is 255 gray levels.

4.4.3 Adaptive Histogram Equalization (local):

The algorithms [103],[104] are used in IPC to enhance the contrast of the grayscale image by transforming the values using contrast-limited adaptive histogram equalization (CLAHE). CLAHE uses the same histogram equalization equations used in section 4.4.2, except in that it splits the image into smaller regions called tiles rather than operating on the whole image as in normal histogram equalization. CLAHE algorithm is an enhancement over AHE algorithms, which has proven to be vulnerable to noise (noise is increased in the resulted image). Pizer et al. , [104] explained that this problem is related to homogenous regions, where noise and pixel intensity levels are close in value. They have shown that limiting the slope in the mapping function can avoid this problem; they also proved that clipping the height of the histogram is equivalent to limiting the slope of the histogram. Based on this, they concluded that CLAHE using contrast limitation can reduce AHE noise enhancement problem.

Figure 4-50 shows the effect on histogram of the image as well as the effect on the image itself. Looking at the image and its histogram, we can see that pixels belonging to gray levels above intensity level 100 have been increased and the histogram of the new image have been smoothed, which indicate local histogram equalization. The adaptive histogram equalization used in IPC accepts four parameters, two of parameters determine the number of tiles to be used for local equalization. This number is advised by Pizer et al. [104] to be between 1/16 to 1/64 of the image size.

The third parameter determines the clip limit which determines the contrast enhancer limits and this should be in the range [0-1]. The fourth parameter determines the desired number of bins in the output image and this is limited to 256.



Original image Enhanced image

Figure 4-50: Effect of local histogram equalization

4.4.4 Fuzzy Enhancement

Tizhosh [105] split fuzzy image processing into three stages: fuzzification, operation on pixels' membership values, and finally defuzzification if it is needed. The actual image processing happens in membership domain instead of the spatial domain.

4.4.4.1 $\lambda -$ enhancement

Lambda enhancement was introduced by Tishoosh et al. [106]. This enhancement algorithm combined Zadeh's definition of membership complement (4-47) with Sugeno's involutive fuzzy complement $\lambda - complement$ (4-48) to introduce a new involutive membership function represented in (4-49)

$$\bar{\mu}(g) = 1 - \mu(g) \tag{4-47}$$

$$\bar{\mu}_\lambda(g) = \frac{1 - \mu(g)}{1 + \lambda\,\mu(g)} \tag{4-48}$$

$$\mu_\lambda^*(g) = \frac{\mu(g)(1+\lambda)}{1 + \lambda\mu(g)} \tag{4-49}$$

Where, gray level g=0,…,L-1 and λ is in the range [-1 ∞]. $\mu(g)$ represents the original membership function.

This new involutive membership function is used to develop a new measure of fuzziness:

$$\gamma(\lambda) = \frac{4}{MN}\sum_{g=0}^{L-1} h(g).\,\mu_\lambda^*(g).\,[1-\mu_\lambda^*(g)] \tag{4-50}$$

λ value that maximizes fuzziness index γ is the optimal value and this can be used to define the new membership function as follows:

$$\mu_{new}^*(g) = \frac{\mu(g)(1+\lambda_{optimal})}{1 + \lambda_{optimal}\mu(g)} \tag{4-51}$$

This algorithm can be summarised as follows:

- Calculate image histogram
- Initialize the membership function $\mu(g)$
- Calculate the index of fuzziness $\gamma(\lambda)$
- Solve the equation $\frac{\partial}{\partial\lambda}\left(\lambda_{optimal}\right) = 0$
- Calculate new membership values using equation (4-51)
- Generate new gray levels $\acute{g} = (L-1).\,\mu_{new}^*(g)$

Figure 4-51shows the results of applying this contrast enhancer on a sample lung X-Rays image.

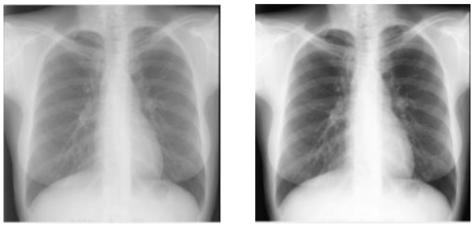


Figure 4-51: Original image (left), lambda enhancement result (right)

4.4.4.2 Fuzzy intensification

This method is also known as minimization of fuzziness Tizhoosh [105]. The goal is reducing the amount of image fuzziness. Based on a chosen membership function, the method first finds the membership value of every pixel in an image. Second step uses an intensifier to transform membership values above a chosen value (default 0.5) to much higher value and the lower member ship values to much lower values. The algorithm can be represented by the following steps:

The first step: Gray-level fuzzification:

$$\mu(g) = \left[1 + \frac{g_{max} - g}{F_d}\right]^{-F_e} \tag{4-52}$$

Where $\mu(g)$ represents the degree of brightness.

Second step is choosing Zaheh’s intensification operator to calculate the new membership values. The intensification operator is represented as follows:

$$\acute{\mu}(g) = \begin{cases} 2[\mu(g)]^2 & if\ \ 0 \leq \mu(g) \leq 0.5 \\ 1 - 2[1 - \mu(g)]^2 & if\ 0.5\ \leq\ \mu(g) \leq 1 \end{cases} \tag{4-53}$$

Finally, the new gray level can be calculated by defuzzification

$$\acute{g} = \ g_{max} - \ F_d\left(\left(\acute{\mu}(g)\right)^{\frac{-1}{F_e}} - 1\right)$$

$$(1 + \frac{g_{max}}{F_d})^{-F_e} \leq \acute{\mu}(g) \leq 1$$

Where F_e and F_d are the exponential and denominational fuzzifiers that control the amount of grayness ambiguity in the membership plane. F_e accepts 1 or 2 as values and F_d can be calculated using the crossover value of the membership $\mu(g_c)$.

Figure 4-52 shows samples of applying this contrast enhancer on lung X-Ray image.

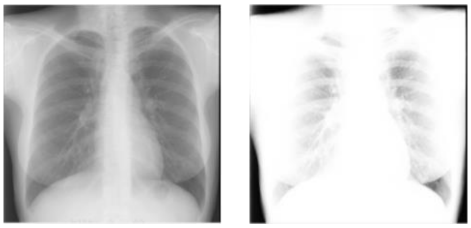


Figure 4-52: Original (left) and enhanced image (right) using fuzzy intensification enhancer

4.5 Algorithms parameters limits

Previous section in chapter 4 explained the algorithms used in our IPC. The DE algorithm will generate parameter values in predefined range specific to every image processing algorithm. These minimum and maximum limits were explained in previous sections for every image processing algorithm. Following tables summarises all limits for every algorithm.

Table 4-1 shows the minimums and maximums for filtering algorithms.

Filter Name	Min Value	Max Value
Median filter	10,10	150,150
Stick 1/2	2,1/2,1	101,3/101,3
Unsharp Masking	0	1
Wiener	2,2	30,30
Average	2,2	50,50
Disk	2	30
Gaussian	0.1,3	50,250
LoG	0.1,3	1,50% img size
Laplacian	0	1
Prewitt		
Sobel		
Morphological 1/2/3/4/5/6	3	2% img size

Table 4-1: Minimum and Maximum for Filters' parameters allowed values

Table 4-2shows minimums and maximums for Contrast Enhancers.

Contrast Enhancer	Min Value	Max Value
Stretching adjuster (*)	0,0,0,0	0.9,1,0.9,1
Global Histogram equalizer	3	5
Local Histogram equalizer		
Local Adaptive Histogram equalizer (**)	2,2,0,100	50,50,1,500
Local Adaptive Histogram equalizer (***)	2,2,0,100	50,50,1,500
Morphological Enhancement	2	50
Fuzzy Complement		
Fuzzy INT	0.1	1

(*) has fifth binary parameter 0 or 1
(**) Rayleigh distribution
(***) Exponential distribution

Table 4-2: Minimum and Maximum Values for Contrast Enhancer algorithms' parameters

Table 4-3 shows minimums and maximums for Thresholding algorithms.

Thresholding Algorithm	Min Value	Max Value
Level thresholding	0	1
Otsu	NA	NA
Kittler	NA	NA
Minimum cross-entropy threshold selection	NA	NA
Triangle threshold		

Table 4-3: Minimum and Maximum values for Thresholding algorithms parameters

Table *4-4* **shows minimums and maximums for post processing algorithms' parameters**

Post Processing Filter	Min Value	Max Value
Filling holes		
Erosion 1/2/3	3,3,3	50,50,50
Dilation 1/2/3	3,3,3	50,50,50
Closing 1/2/3	3,3,3	50,50,50
Opening 1/2/3	3,3,3	50,50,50

* disk shape of four periodic lines
** disk shape of six periodic lines
*** disk shape of eight periodic lines

Table 4-4: Minimum and Maximum for Post Processing Algorithms parameters

Above mentioned maximums and minimums are very critical to the correct functioning of the DE optimizer. Every new generation, and due to the DE chosen strategy, the population members' variables could go above or below allowed limits. This requires adding a correction logic to DE optimization loop, where parameters are compared with maximum and minimum, and reset any variable that goes beyond the limit of the border that was exceeded

4.6 Summary

In this chapter, we surveyed all the algorithms that have been used in our image processing chain. For every algorithm, we have explained its theory and illustrated the effect of applying it on sample images. We have also identified algorithms' parameters and developed their maximums and minimums either by experiment and visual observation, or by knowing the parameters' physical limitations.

Chapter 5

Image Processing Chain Framework

5.1 Proposed Image Processing Chain (IPC)

The image processing chain IPC framework defines the sequence of filters, contrast enhancers, thresholding, and post processing algorithms. The number of algorithms in this IPC is restricted to ten algorithms by design. The algorithms are chosen from the pool of algorithms explained in the previous chapter. Chapter 4 listed only algorithms that are coded and pooled for DE optimizer usage, but scientific literature has a large number of algorithms that could be used in preparation for binarization of image and post processing after binarizing the image. DE optimizer attempts to find the best sequence of algorithms and their optimal parameters that should be applied for specific task and image modality to get optimal results. This universal algorithm has sixty-four dimensions.

The goal of the framework as it is designed is to target images that could be split into two classes background and foreground (which is in our case the tissue to be segmented). This requires gold images of tissues that are the target of segmentation to be able to calculate the best chain. The framework design does not target images with one class of pixels (i.e. images with no features such as lung, cancer, tumor, heart, would defect, metal cracks, etc.)

Scientific literature suggested many segmentation algorithms as discussed in the previous chapter. These algorithms didn’t invest much in preparing the image and enhance it as such the targeted tissue pixels becomes easier to extract from the background. The results in this thesis shows how it is critical to prepare the image before segmenting any of the thresholding techniques. The problem here lies in finding the best sequence of filters and best parameters’ values for these algorithms to optimally prepare the image for the optimal thresholding technique.

The dimensionality of the problem and the huge number of algorithms we can put in the pool makes it impossible to create IPC even by experts and suggest optimization methods to solve this problem.

Enhancing the image requires passing it through an optimized IPC, which is a series of filters and contrast enhancers, before applying thresholding. Then, the thresholding result is passed to post processing filters to fill holes and make sure that the segmented contour is smooth. Our suggestion, the image processing chain, is impossible to be solved by engineers and experts. It involves electing the right series of filters, enhancers, thresholding, and finally post processing algorithms out of hundreds of algorithms, and fine-tune related parameters.

This thesis proposes The DE optimization algorithm to create the image processing chain; at the same time, the optimizer fine-tune the chosen chain algorithms' parameters to target a specific tissue in images produced by specific imaging modality. Thus, the produced image processing chain with its fine-tuned parameters can be trained to segment the same tissue in new images from the same modality.

Figure 5-1 shows the suggested chain with algorithm pools used in each stage. The lower part shows the training phase. DE optimizers accept three training images with their Golds and a pool of all algorithms in all phases.

DE optimizer creates the appropriate and tuned image processing chain for the provided specific tissue and specific imaging modality. DE will generate randomized population members (black box optimization) with values ranging between minimum and maximum values (box constraints optimization problem). We have a chromosome with sixty-four mixed type parameters (mid-scale optimization problem) which increases when we add new algorithms to the pools of IPC, making the training phase extremely time consuming. On the other hand, using the created and tuned IPC to segment new images is very fast. This approach permits creating a new chain for every tissue and modality we need and loading the appropriate IPC to segment needed tissue, which will be illustrated in the following section.

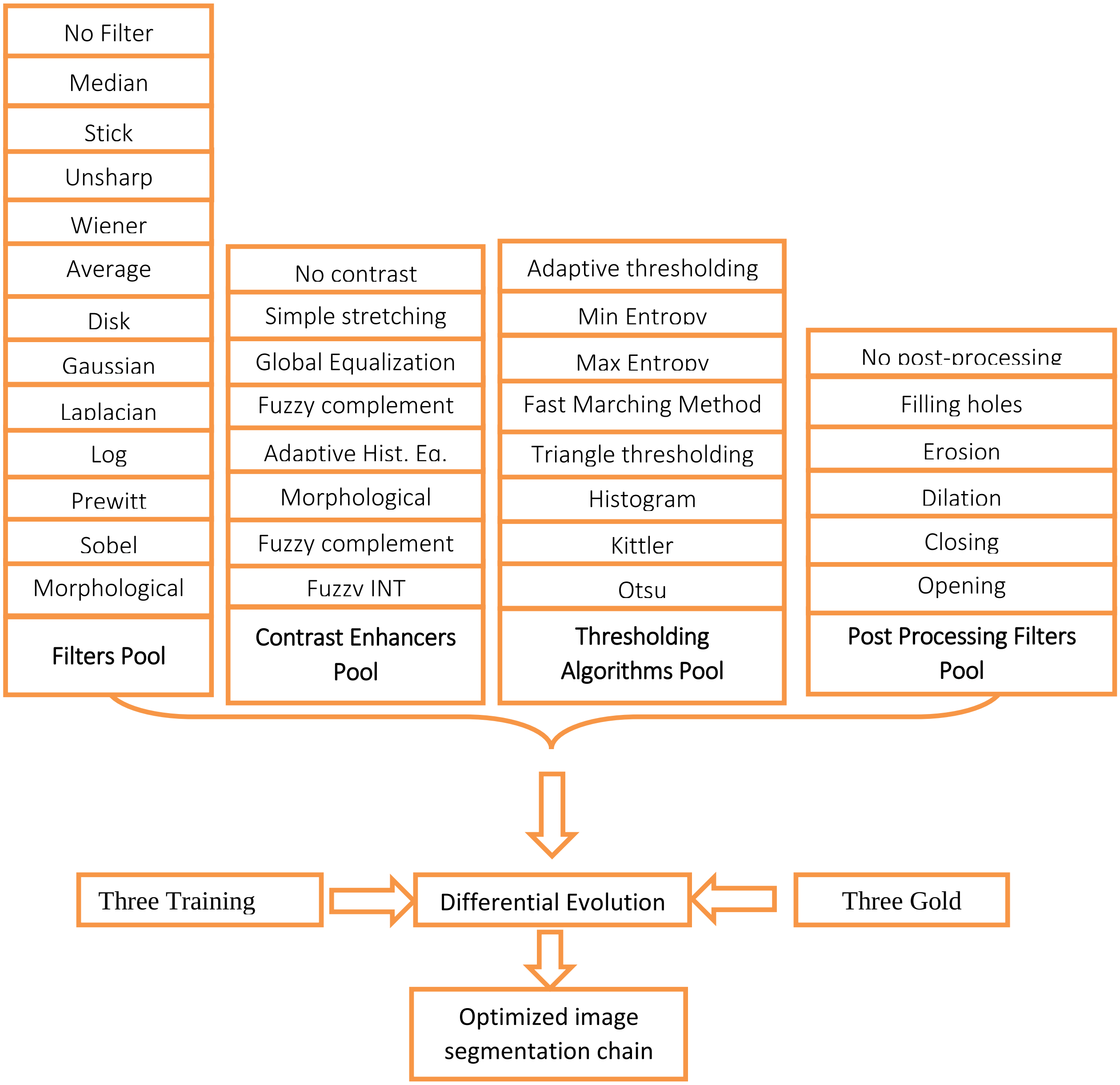


Figure 5-1: Illustration of Image Optimization Chain and algorithms used by the DE optimizer

5.2 Building Image Processing Chain Chromosome

The main target of the image processing chain is preparing the image to be segmented optimally, then passing the output image to post processing to enhance the segmentation results. There are limitless chain designs and limitless number of algorithms that can be added to the pool of algorithms used by DE. The only restriction that limits IPC design complexity is the CPU power available to solve the increasing number of dimensions which is caused complex IPC designs and higher number of algorithms added to the pool.

Due to the limited CPU processing power available to us we decided to design an Image Processing Chain composed of only ten stages, which are explained below. The DE optimizer chooses ten algorithms out of the pool to form the IPC and fine tune their parameters. These ten algorithms are sequenced as illustrated in Figure 5-2.

The algorithms, their sequence and parameters values are chosen by DE which makes it essential to design DE chromosome to enable the DE for creating this IPC. DE population chromosome is designed as follows:

- First ten variables are integer variables and they decide the chosen algorithms, their sequence and number of filters, contrast enhancers, thresholding algorithm, and post processing algorithms.
- The following fifty-four variables in the chromosome are the real type and they represent algorithms' parameters arranged in a respected order throughout the program, which makes it easy to extract in sub-functions.

The full chromosome structure is designed as follows:

Population member = [Chain sequence (ten variables), Filters algorithms' parameters, Contrast algorithm Parameters, Post Processing algorithms' parameters, Thresholding algorithm parameters]

This chromosome combines every filter, contrast enhancer, thresholding, and post processing algorithm in the pool, which adds complexity to our solution. The chromosome contains discrete and real variables in addition to variables contained in a limited set which they should be defined and constrained.

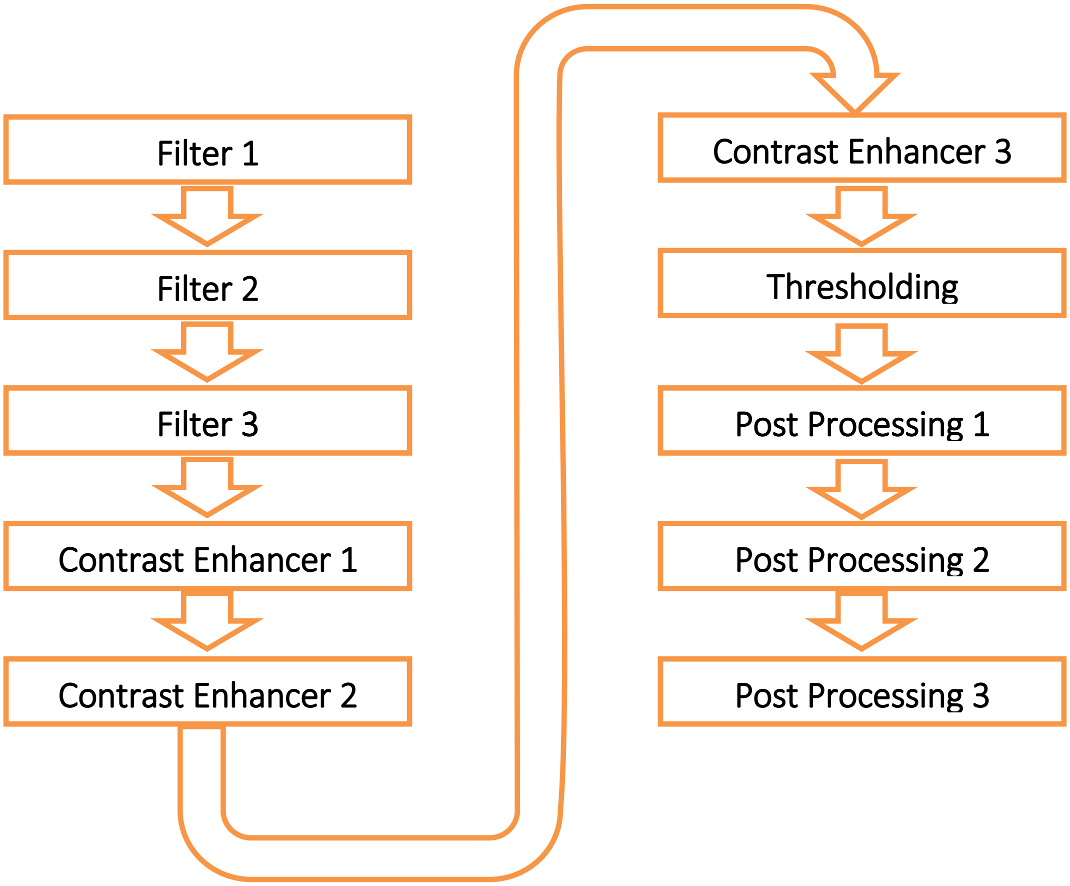


Figure 5-2: DE built IPC segmentation sequence

5.3 Building Algorithm Pools

In order to minimize memory needs and decrease parameters passing mechanism between functions, the pool is structured to have a chain function for each stage in the proposed image processing chain as follows:

- chain_filters
- chain_contrats
- chain_thresholding
- chain_postprocssing

Each chain function is composed of switch statement that list all algorithms of the pool. These chain functions need two vectors to work properly:

- Ten chain sequence parameters which identify the algorithms that should be called from the pool. The chain function selects its own three sequence parameters out of ten and calls the proper functions. The Image processing chain accepts only one thresholding algorithm which explains why we have ten sequence parameters (three for each chain pool and one for thresholding).
- Parameters values vector belonging to the pool of algorithms that it calls. Each algorithm in the chain function knows its own parameters in the parameter vector and can access it directly from the pool.

Below is a sample pseudo-code of chain_filter function responsible for filters pool, which is the first stage of the image processing chain.

```
Pseudo Code for Chain_Filter function
        Get filter to be applied
        Get filter parameters from parameter vector
        Apply filter with related parameters
        Return filtered image to DE for further processing
End
```

5.4 DE parameter Settings

DE parameters were adjusted to target this problem. Table 5-1 shows DE parameter settings. Stopping criteria was based on the number of generation as shown in Table 5-1. Every generation DE evaluates every population member and minimize the output of a fitness function built specifically to target this problem. This fitness function compares segmented image using population member with gold image that was segmented by

PARAMETER	VALUE
Number of Generations	5000
Population size	500
Crossover probability	0.9
DE step size	0.5
DE strategy	DE/rand/1/bin

Table 5-1: Differential Evolution optimizer parameter settings

physician according to the following formula:

$$f(x) = \sum_{img=1}^{k} \sum_{m=1}^{sm} \sum_{n=1}^{sn} (Image\ \ xor\ Gold) \tag{5-54}$$

Where k is the number of training images while sm and sn are the image size in both dimensions of the image. This fitness function counts the number of incorrectly classified pixels in every segmented training image and deliver this number to DE to be used for minimization.

The fitness function (5-54) designed for this problem allow the addition of more training images. Every new training image added will force DE to evaluate all population members which will increase the elapsed time needed by each generation (i.e. doubling the training image will double the evaluation time for every generation).

The fitness function is designed to loop through all training image without the need for any modification on the program.

Even though the chain is composed in our solution of defined stages namely, filtering, contrast enhancing, thresholding, post processing; we still can’t split the fitness function. There are intermediate gold images in between stages, also every stage is dependent on the output of the stage before.

5.5 Image Processing Chain parameters’ indexes

Every algorithm in the pool has multiple parameters that have a certain location in the chromosome vector. These locations are strictly adhered to throughout the program design. The below tables list the parameters of every algorithm and the related indexes as they appear in the chromosome.

Filters’ vector details as shown in Table 5-2 lists filters’ parameters, which consists of twenty-five variables indexed one to twenty-five.

Filter Name	Param. index in chromosome
Median filter	1,2
Stick 1/2	3,4/ 5,6
Unsharp Masking	7
Wiener	8,9
Average	10,11
Disk	12
Gaussian	13,14,15
Log	16
Laplacian	17,18,19
Prewitt	No parameter
Sobel	No parameter
Morphological 1/2/3/4/5/6	19,20,21,22,23,24,25

Table 5-2: Filters parameters location in the chromosome (Index)

Table 5-3: **Contrast enhancers parameters location in the chromosome (Index)** shows Similar to filters,contrast enhancer algorithms with their parameters' indexes.

Contrast enhancer	Param. Index in chromosome
Stretching adjuster	1,2,3,4
Global Histogram equalizer	5
Local Histogram equalizer	No Parameters
Local Adaptive Histogram equalizer (*)	6,7,8,9
Local Adaptive Histogram equalizer (**)	10,11,12,13
Morphological Enhancement	14
Fuzzy Complement	No parameters
Fuzzy INT	15

(*) Reyleigh distribution

(**) Exponential distribution

Table 5-3: Contrast enhancers parameters location in the chromosome (Index)

Contrast enhancer parameters are gathered in contrast vector. The Table 5-3 shows fifteen enhancer used in our IPC with its related parameters' indexes in the contrast vector.

We have four post processing filters. These processing filters are essential to improve the final binary image by filling holes, closing, opening, and dilation which has proved to be very effective in such binary image to improve the final segmentation results. Each filter of these can be applied using different disk shape, with four, six, or eight periodic lines, as shown in the Table 5-4.

Post Processing Filter	Param. Index in chromosome
Filling holes	No parameters
Erosion 1/2/3	1*,2**,3***
Dilation 1/2/3	4*,5**,6***
Closing 1/2/3	7*,8**,9***
Opening 1/2/3	10*,11*,12***

* disk shape of four periodic lines
** disk shape of six periodic lines
*** disk shape of eight periodic lines

Table 5-4: Post Processing algorithms' parameters' location (index)

5.6 Summary

In the first part of this chapter, we introduced an image processing chain designed to combine four pools of algorithms; each pool provides the DE optimizer with algorithms necessary to pass the image through different stages of the IPC. In the second part, we introduced the solution to structure the DE chromosome for sixty-four algorithms existing in different pools. We also explained the necessity of maximum and minimum vectors for correcting the generated population members in DE. Finally, we introduced DE optimizer settings and provided a detailed listing of every algorithm used in every pool and its index in the chromosome vector, in addition to every parameter's maximum and minimum values.

Chapter 6

Case Studies and Discussions

6.1 Case Study: Segmenting Lungs X-Ray images

Lungs' X-Ray images are one example of medical images that are poorly illuminated and they contain many overlapping structures (i.e., chest cavity bones, hear, etc.). These overlapped structures have very close intensity levels as the lungs, which makes finding the correct enhancers and thresholding parameters very difficult. In the following section, we will discuss a full cycle of training the framework, testing, verification of training, and finally, we will discuss the results.

6.1.1 Training Phase

The framework training requires three images with their gold images, which are segmented lungs by experts (Figure [6-1]). Selecting these training images proves to be critical for the learning process. They should represent extreme situations as well as normal for the imaging modality used to capture these images. In the X-Ray lung images case we chose high, low, medium contrast and edge visibility as visual criteria.

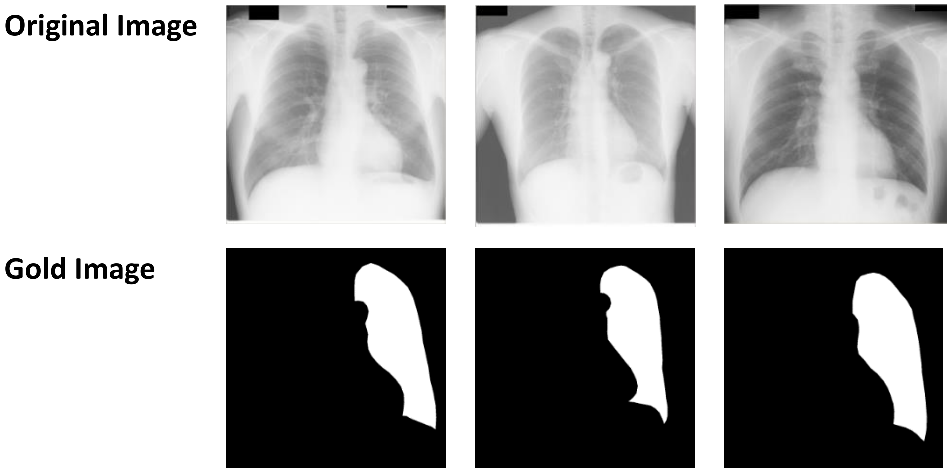


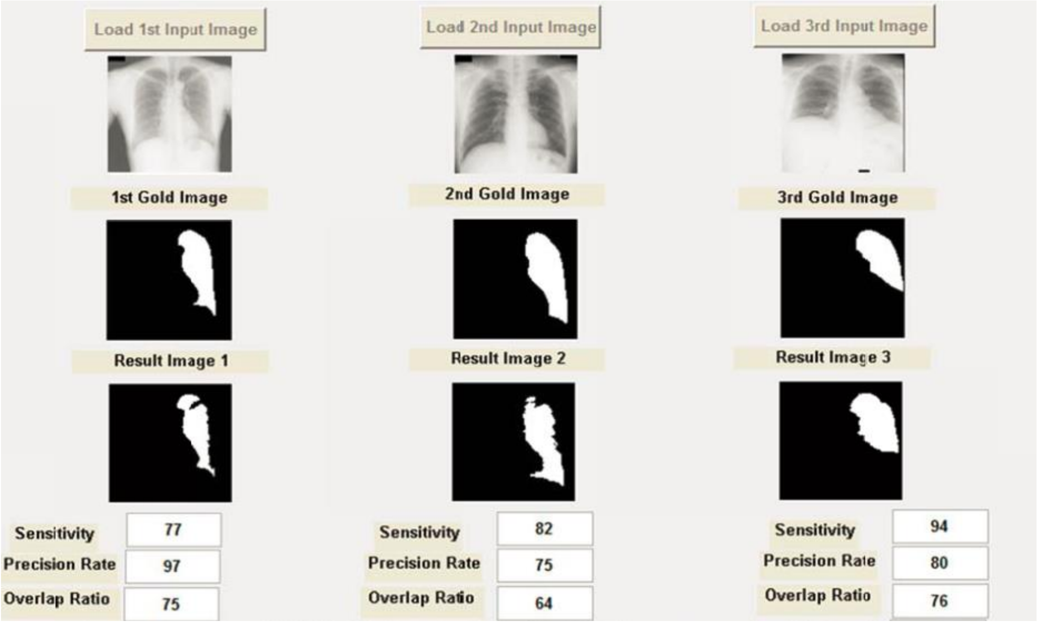
Figure 6-1: Three training images with their respective gold images

The framework can accept gold images with two lungs. There is no modification needed on the code or the logic of the framework. We used classical parameter settings as shown in table 6-1 with 5000 maximum generation as a stopping criteria.

Figure 6-2 shows a screen shot of the program during training phase. The first and second row of images are the loaded training samples with their gold images, while the third row shows the segmentation results of best population member found till the current generation.

Number of Generations	5000
Population size	500 (ten times problem dimension)
Crossover probability (CR)	0.9
DE step size	0.5
DE strategy	DE/rand/1/bin

Table 6-1: DE parameter settings for lung segmentation



Sensitivity 77
Precision Rate 97
Overlap Ratio 75

Sensitivity 82
Precision Rate 75
Overlap Ratio 64

Sensitivity 94
Precision Rate 80
Overlap Ratio 76

Figure 6-2: Program GUI interface during training phase

The program provides multiple indicators of the quality of segmented image.

The following definitions and metrics (i.e., precision, sensitivity, and overlapping ratios) are utilized to report the numerical results of our experiments.

- True positive (TP): Tissue pixel (foreground) is correctly diagnosed as tissue pixel.
- False positive (FP): Non-tissue pixel (background) incorrectly identified as tissue pixel.
- True negative (TN): Non-tissue pixel correctly identified as Non-tissue pixel.
- False negative (FN): Tissue pixel incorrectly identified as Non-tissue pixel.
- Image Pixels (ImP): Represents the total pixels existing in the processed image.

$$\text{Precision} = TN/(TN + FP) \tag{6-55}$$

$$\text{Sensitivity} = TP/(TP + FN) \tag{6-56}$$

$$\text{Overlap} = (TP + TN)/ImP \tag{6-57}$$

All three segmented results are combined in the fitness function, which mandates choosing the three training images to represent the whole range of possibilities for the X-Ray modality. Different brightness, contrast level, and shape provide the optimizer with better representation of lung tissues. These variances yield a better trained IPC and higher accuracy in later segmentation of new Lung X-Ray images when use the trained IPC.

In the following **Error! Reference source not found.**, we can find how the optimizer develops the image processing chain for better segmentation. Initialization starts with a calculation of the centroid of the provided gold image to get the seed point that determines whether the tissue needs to be segmented, which explains why the second generation has a white dot showing the segmentation results.

On the sixty-third generation, where each generation represents five hundred population members, we can see the first real approximation of lung tissues.

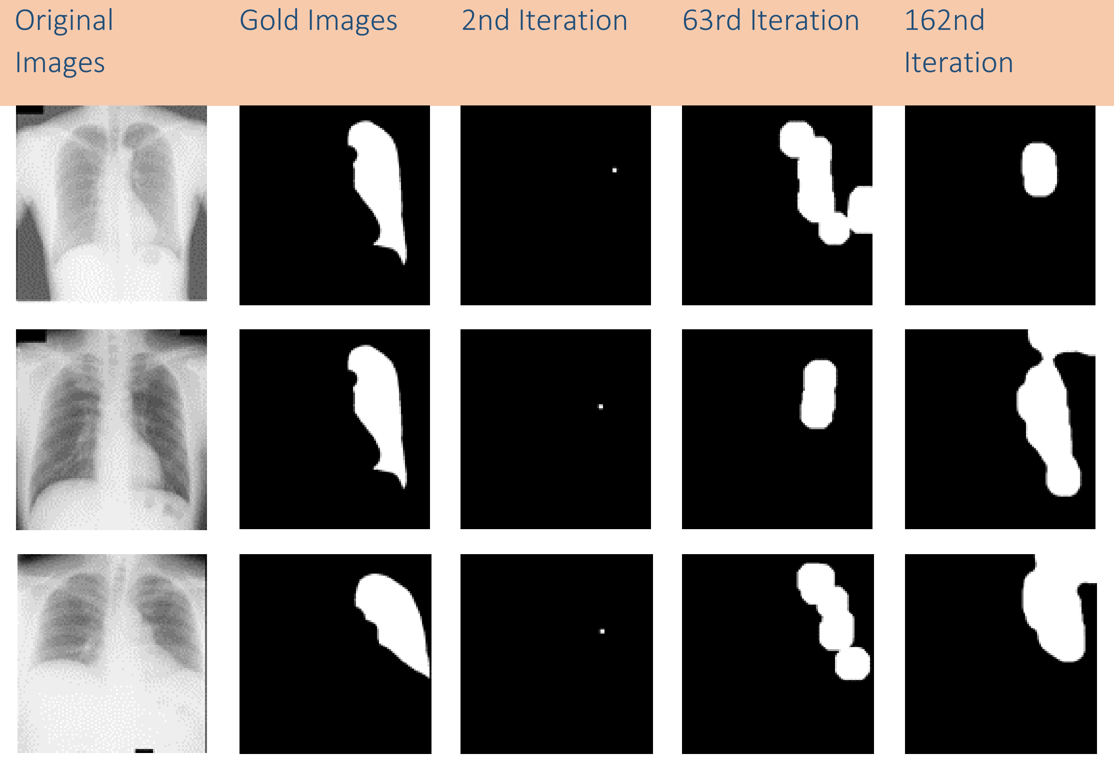


Figure 6-3: Segmentation results by IPC during generations 2,63, and 162

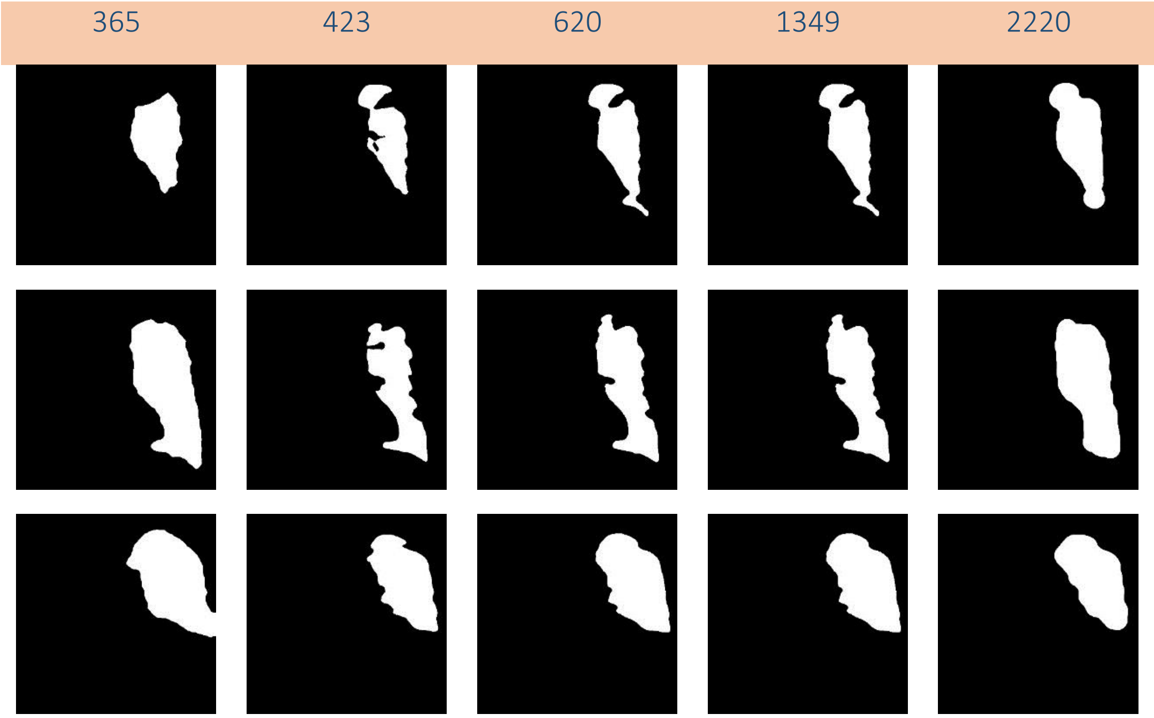


Figure 6-4: Segmentation results by IPC during generations 365,423,620,1349, and 2220

Previous visual illustration of convergence can also be expressed in terms of fitness function value and iteration number (Figure 6-6).

This convergence diagram shows that by iteration six-hundred, there are five improvements on the fitness value, then the optimizer hits a plateau that lasts for almost two-thousand iterations before we get a new improvement.

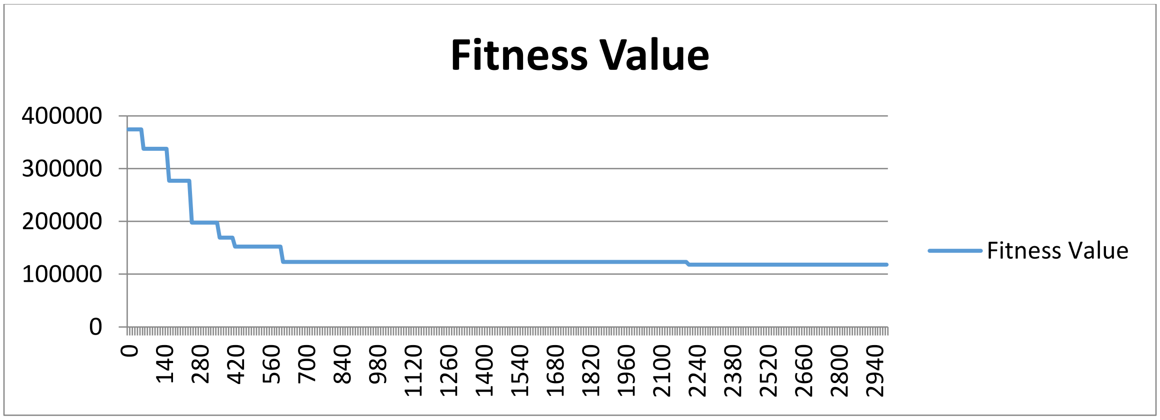


Figure 6-5: Fitness value vs Generation

The complication of the chromosome design has mandated that each population contains all the parameters for all algorithms in all pools.

Every population member elects ten algorithms out of forty-four, which means we are using the parameters for only ten algorithms. As a consequence, the optimizer takes longer to exhaust a solution space. The output of this phase is a vector of sixty-four elements that specifies the sequence of elected algorithms of the Image Processing Chain, followed by the parameters of these elements.

6.1.2 Optimal Image Processing Chain

Each algorithm in the pool delivered to DE optimizer is tagged with an identity number.

Chromosome Index	1	2	3	4	5
Algorithm Type	Filter	Contrast	Post	Threshold	filter
Algorithm Identity	8	3	1	0	4
Algorithm Name	Gaussian	HistEq	Fill Holes	Otsu	Weiner

Parameter Index	6	7	8	9	10
Algorithm Type	Contrast	Post	filter	contrast	Post
Algorithm Number	7	2	8	2	0
Algorithm Name	Fuzzy complement	Fill hole	Gaussian	Hyperbolization	No

Table 6-2: Image Processing Chain as created by DE

During optimization, DE generates a number between the minimum and maximum value (based on the number of algorithms in the pool) that selects one of these algorithms based on their identity number. The first ten elements in the chromosome vector specify these elected algorithms by DE optimizer, and the values of each cell specify an algorithm identity number. Table 6-2 shows the algorithms' names that match the algorithms' identity values in the chromosome. The parameters needed by these algorithms have higher indexes in the optimal chromosome (64 cells) and is showed in the next section "Tuned Parameters".

6.1.3 Optimal Parameters

Table 6-3 to Table 6-9 list the final optimal values of the chromosome vector. These values are linked to every algorithm in the pool as specified in:

- Filter algorithms Table 5-2 (Chromosome indexes 11-35)

- Contrast algorithms Table 5-3 (Chromosome indexes 36-40)
- Post processing algorithms Table 5-4 (Chromosome indexes 41-54).

Chromosome vector index	11	12	13	14	15	16	17	18
Parameter Value	12.5	6.18	27.13	1.11	8.16	2.6	0.61	17.22

Table 6-3: Algorithms' parameters stored in optimized chromosome indexes 11-18

Chromosome vector index	19	20	21	22	23	24	25	26
Parameter Value	3.96	4.72	24.21	10.25	7.45	7.18	5.6	0.78

Table 6-4: Algorithms' parameters stored in optimized chromosome indexes 19-26

Chromosome vector index	27	28	29	30	31	32	33
Parameter Value	16.64	23.52	423.66	321.76	374.74	195.7	146.43

Table 6-5: Algorithms' parameters stored in optimized chromosome indexes 27-33

Chromosome vector index	34	35	36	37	38	39	40
Parameter Value	168.54	156.41	0.6	0.97	0.68	0.7	48.97

Table 6-6: Algorithms' parameters stored in optimized chromosome indexes 34-40

Chromosome vector index	41	42	43	44	45	46	47	48
Parameter Value	28.62	38.27	0.27	440.15	31.27	25.91	0.44	175.46

Table 6-7: Algorithms' parameters stored in optimized chromosome indexes 41-48

Chromosome vector index	49	50	51	52	53	54	55	56
Parameter Value	43.89	0.18	38.77	3.22	6.63	9.05	47.09	39.16

Table 6-8: Algorithms' parameters stored in optimized chromosome indexes 49-56

Chromosome vector index	57	58	59	60	61	62	63	64
Parameter Value	28.7	40.53	29.53	45.4	42.65	42.18	12.34	33.4

Table 6-9: Algorithms' parameters stored in optimized chromosome indexes 57-64

6.1.4 Testing Phase

Once the image processing chain is created and optimized; it can be used to segment tissues in new images. Two below restrictions should be taken into account.

- The new image should be produced by the same modality.
- The tissue should be same with the tissue that the IPC was trained on it.

In order to verify the performance of the training stage, we used an image of database that contains 247 lung X-Ray images with their gold images. This database is available online

[107].Detailed results with all three parameters' calculation for all 247 lung X-Ray images is listed in Appendix A.

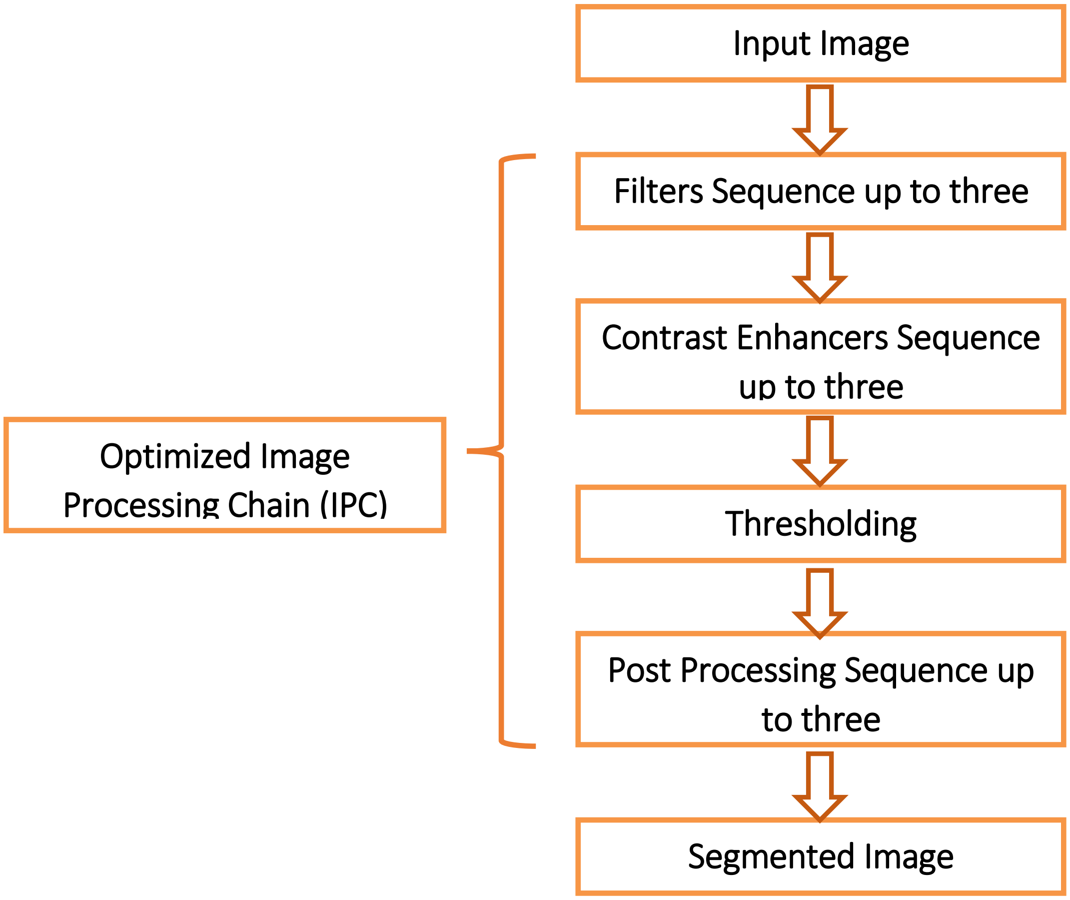


Figure 6-6: Segmentation using trained IPC

6.1.5 Results analysis

The optimized image processing chain contains eight thresholding algorithms. Using each of these algorithms as a standalone thresholding and segmentation methods yields to a very poor performance as shown in Table 6-10.

The optimized chain has almost seven times higher accuracy that just normal thresholding technique. This accuracy increases by adding more filters and enhancers to the IPC pools. The last row in Table 6-10 verifies the optimality of the developed IPC by changing one algorithm in the optimal chromosome. The obtained average shows the severe effect on the trained system if it is manually manipulated. Detailed result for every image in the database can be found in Appendix C.

Thresholding Algorithm	Overlap with Gold percentage	Image Database
Otsu	13%	247 Lung X-Ray
Kittler	12.48%	247 Lung X-Ray
Histogram	7.79 %	247 Lung X-Ray
Triangle	10.28%	247 Lung X-Ray
Image Processing Chain	72%	247 Lung X-Ray
Image Processing Chain after manual change of one algorithm	47.06%	247 Lung X-Ray

Table 6-10: Segmenting images using individual thresholding algorithms used in IPC

6.2 Case Study: Segmenting of Breast Cancer Ultrasound images

Breast cancer segmentation is a very challenging task due to inherent speckle noise and low contrast of breast ultrasound imaging.

6.2.1 Training Phase

Below, we can see the three images used for training the image processing chain. They are chosen to represent the most possible images for the trained IPC in the production.

The first image (top left) shows a sample of images with very low contrast where targeted lesion and background have very close intensity levels. The second image shows a high contrast image with clear cancer lesion borders and a small amount of speckle noise. The third image on the top right shows high contrast with speckle covering cancer lesion border.

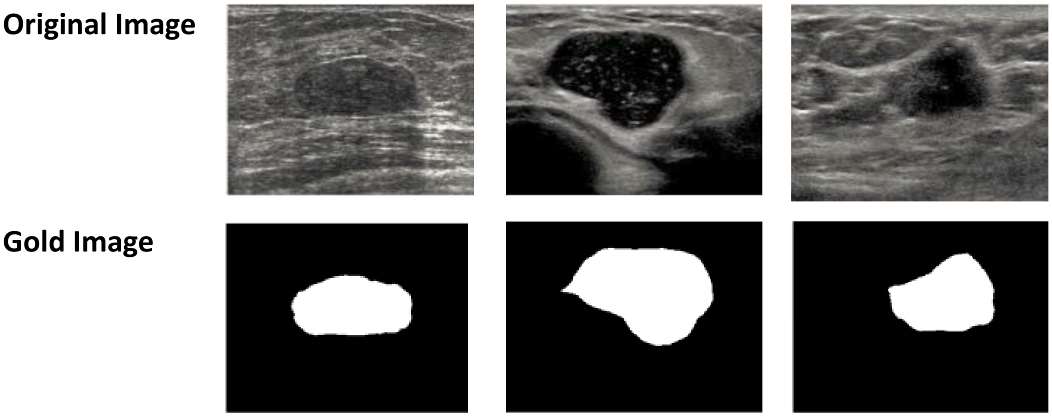


Figure 6-7: Three training images for breast cancer

The second row shows gold images segmented by an expert (physician). Similar to Lung X-Ray images, the framework can be trained on multi tumor extraction; We still have to provide gold images with segmentation of all needed tissues to be extracted.

Number of Generations	5000
Population size	500 (ten times problem dimension)
Crossover probability (CR)	0.9
DE step size	0.5
DE strategy	DE/rand/1/bin

Table 6-11: DE parameter settings for Lung segmentation

The DE optimizer is set using classical parameter settings same as in the lung X-Ray.

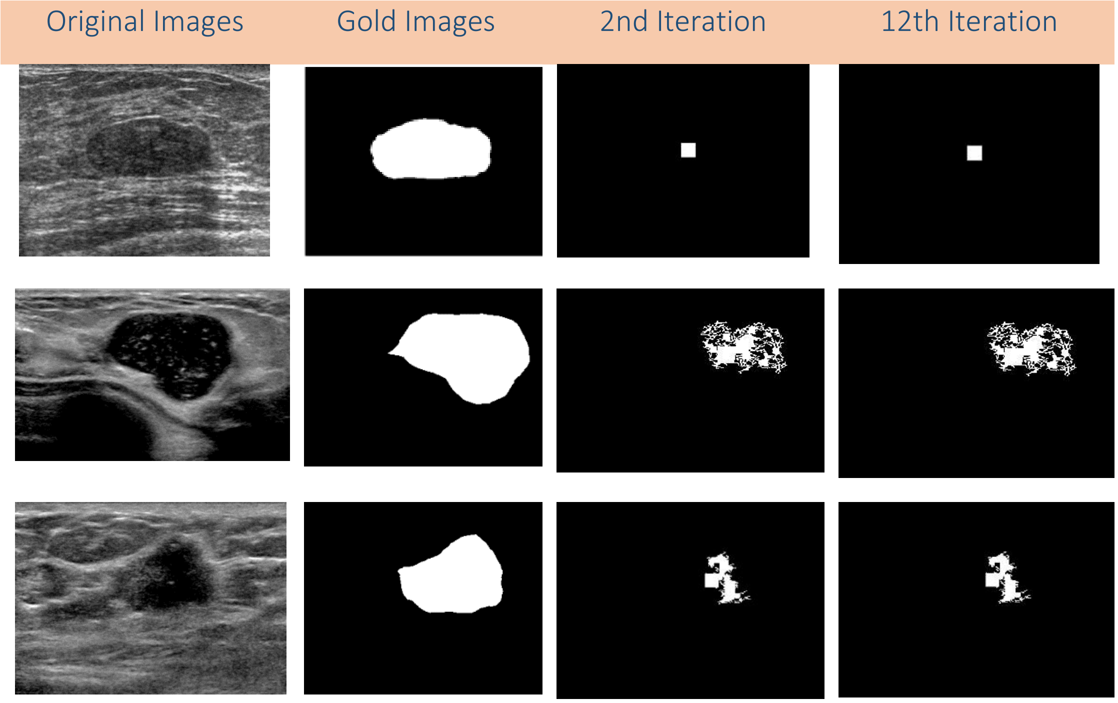


Figure 6-8: Segmentation results for 2nd and 12th generation

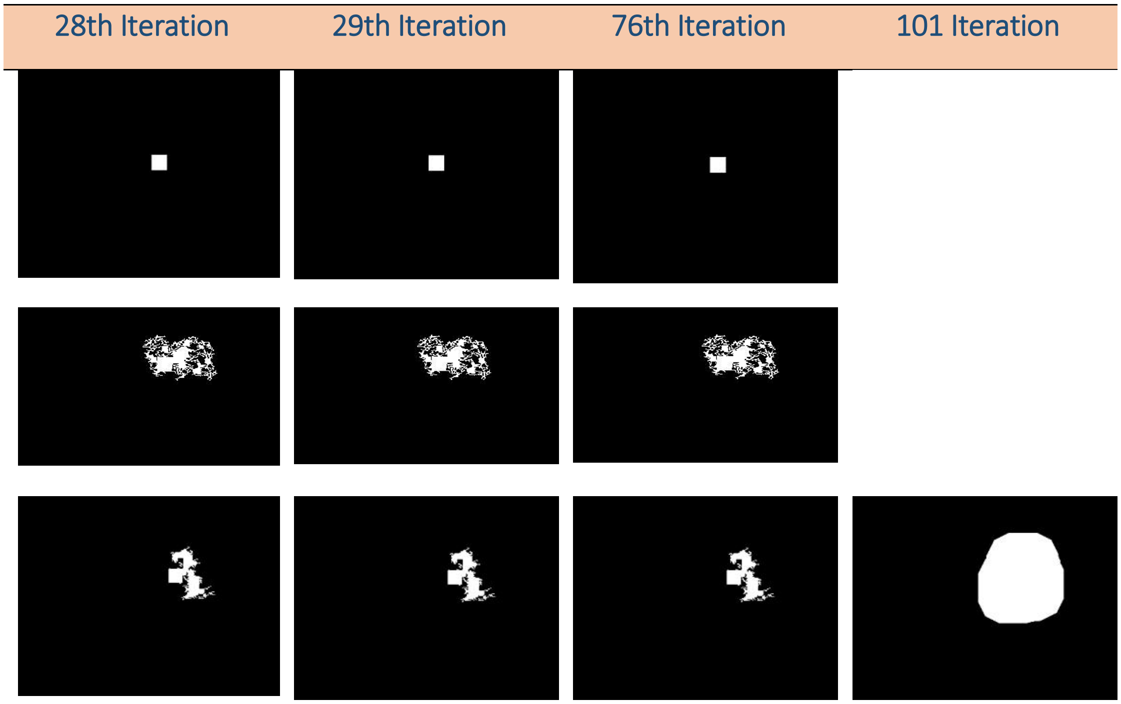


Figure 6-9: Segmentation results for 28th,29th,76th, and 101 generations

In the 410 generation, each generation represents five hundred population member, we can see the first real approximation of breast cancer tissue.

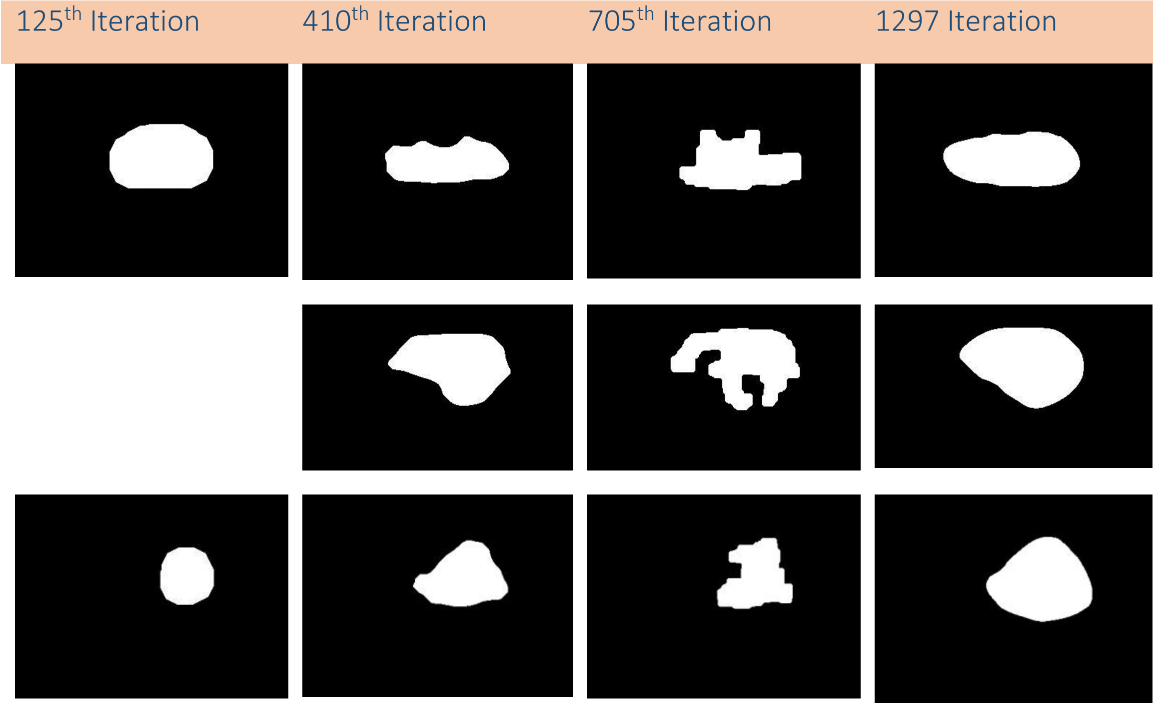


Figure 6-10: Segmentation results for 125th,410th,705th, and 1297 generation

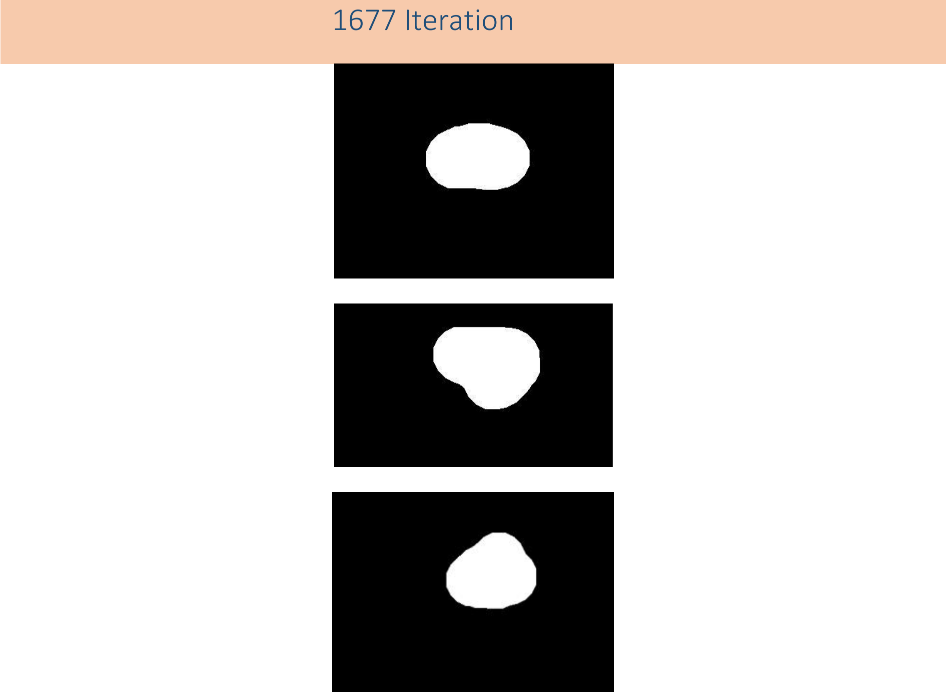


Figure 6-11: Segmentation results for 1677th generation

Finally, in generation 1677th (Figure 6-11), we can see the final best Image Processing Chain.

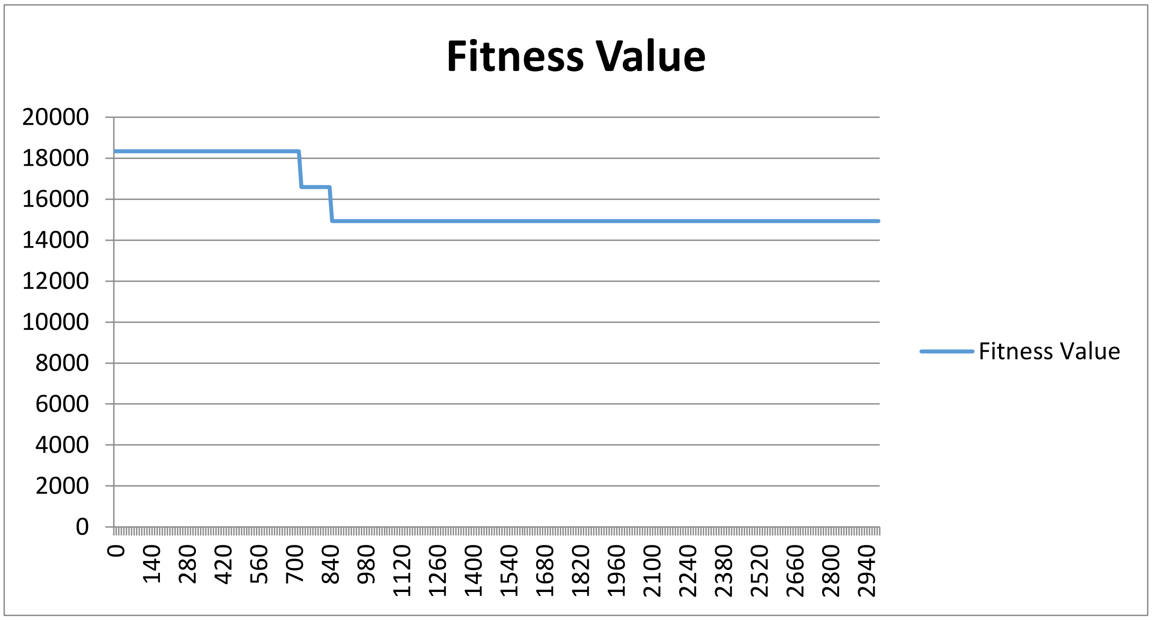


Figure 6-12: Fitness vs Generation

DE optimizer uses the overlap between the gold image (physician segmentation) and IPC resulted segmentation as a fitness function. Figure 6-12 shows the relationship generation vs fitness value. The graph shows that this is a minimization problem (we are trying to minimize number of lung pixels mistakenly classified as non).

6.2.2 Optimal Image Processing Chain

Each algorithm in the pool delivered to DE optimizer is tagged with an identity number. During optimization, DE is generated a number between minimum and maximum value (based on the number of algorithms in the pool) that selects one of these algorithms based on their identity number. The first ten elements in the chromosome vector specifies these elected algorithms by DE optimizer, and the values of each cell specify an algorithm identity number.

Table 6-12 shows the algorithms' names that matches the algorithms' identity values in the chromosome. The parameters needed by these algorithms have higher indexes in the optimal chromosome (64 cells), and is showed in the next section, "Optimal Parameters".

Chromosome Index	1	2	3	4	5
Algorithm Type	Filter	Contrast	Post	Threshold	Filter
Algorithm Identity	3	8	1	0	1
Algorithm Name	Stick	FuzzyINT	Fill Holes	Otsu	Stretching

Parameter Index	6	7	8	9	10
Algorithm Type	Contrast	Post	Filter	Contrast	Post
Algorithm Number	6	14	0	7	0
Algorithm Name	Morphological	Opening	No	Fuzzy compl.	No

Table 6-12: Image Processing Chain as created by DE

6.2.3 Optimal Parameters

Table 6-13 to Table 6-18 list the final optimal values of the chromosome vector. These values are linked to every algorithm in the pool as specified in:

- Filter algorithms Table 5-2 (chromosome indexes 11-35)
- Contrast algorithms Table 5-3 (Chromosome indexes 36-50)
- Post processing algorithms Table 5-4 (Chromosome indexes 51-54).

Chromosome vector index	11	12	13	14	15	16	17	18
Parameter Value	15.3	19.5	35.3	1.8	2	2.9	1	21.9

Table 6-13: Algorithms' parameters stored in optimized chromosome indexes 11-18

Chromosome vector index	19	20	21	22	23	24	25	26
Parameter Value	25.9	11.5	13.1	8	2	5.3	14.1	0.6

Table 6-14: Algorithms' parameters stored in optimized chromosome indexes 19-26

Chromosome vector index	27	28	29	30	31	32	33
Parameter Value	30	25.8	1000	83	1	299.3	193.7

Table 6-15: Algorithms' parameters stored in optimized chromosome indexes 27-33

Chromosome vector index	34	35	36	37	38	39	40
Parameter Value	4.8	390.4	0.1	0.7	0.6	0.7	3

Table 6-16: Algorithms' parameters stored in optimized chromosome indexes 34-40

Chromosome vector index	41	42	43	44	45	46	47	48
Parameter Value	50	2	0.2	500	50	22.7	0.6	122.9

Table 6-17: Algorithms' parameters stored in optimized chromosome indexes 41-48

Chromosome vector index	49	50	51	52	53	54	55	56
Parameter Value	49.2	0.4	9.1	50	50	50	7.5	46.2

Table 6-18: Algorithms' parameters stored in optimized chromosome indexes 49-56

Chromosome vector index	57	58	59	60	61	62	63	64
Parameter Value	24.1	37.2	10.4	11.4	5.9	10.6	3	43.2

Table 6-19: Algorithms' parameters stored in optimized chromosome indexes 57-64

6.2.4 Testing Phase

Similar to testing phase in section 6.1.2, once the Image Processing chain is created and optimized for breast cancer images, we can use it to segment new images. Figure 6-6 shows the stages that any new image passes through to get segmented.

The segmentation program uses the trained IPC to conduct new breast cancer segmentation. First step, as shown in Figure 6-6, is loading the optimized IPC. In order to verify the performance of the training stage, we used an image of database that contains 30 breast cancer images with their gold images. Detailed results with all three parameters' calculation for all 30 breast cancer images are listed in Appendix B.

6.2.5 Result Analysis

The optimized image processing chain contains eight thresholding algorithms. Using each of these algorithms as a standalone thresholding and segmentation methods yields to a very poor performance, as shown in Table 6-20. The optimized chain has almost seven times higher accuracy that the normal thresholding technique.

This accuracy increases by adding more filters, enhancers, thresholding, and post processing algorithms to the IPC pools to be used by DE. Last row in Table 6-20 verifies the optimality of the developed IPC by changing one algorithm in the optimal chromosome. The obtained average shows the severe effect on the trained system if it was manually manipulated (detailed result for the whole images database can be found in Appendix D.

Thresholding Algorithm	**Overlap with Gold**	**Image Database**
Otsu	13.7%	30 Breast images
Kittler	17.24%	30 Breast images
Histogram	13.85 %	30 Breast images
Triangle	20.79%	30 Breast images
Image Processing Chain	52.37%	30 Breast images
Image Processing Chain after manual change of one algorithm	7.2	30 Breast images

Table 6-20: Segmenting images using individual thresholding algorithms used in IPC

Chapter 7

Conclusion and Future work

7.1 Conclusion

The surveyed scientific literature for optimization-based segmentation algorithms suggested many segmentation algorithms that target specific type of images and modality; the optimizers in these algorithms target one specific task without taking into consideration the whole image segmentation process. The proposed approach in this thesis attempts to consider the gap in scientific literature by proposing a universal segmentation framework that is trained to segment any feature from images produced by any modality. Unlike expert systems, where you need a huge amount of data to get the system trained properly, our framework uses only three images with their related gold images to train the system. Once the training is finished, the trained IPC is loaded and re-used for the segmentation of the same type of features.

The novelty of our approach is that the DE optimizer creates a full image processing system composed of filters, contrast enhancers, thresholding, and post processing. It also tunes all parameters of these algorithms in such a way that the IPC is able to segment the targeted features. To the best of our knowledge, this is the first attempt to target segmentation as a fully integrated process where image passes through a sequence of processing steps to reach the final segmented status.

7.2 Future work

The population size for this type of problems is five-hundred members with a dimensionality of sixty-four. Evaluation of one generation takes about a day or two for certain value of algorithm values. Also some of the algorithms in the pool are costly in terms of time. Achieving acceptable results need five-thousand iterations and finish for three months as in both cases discussed in Chapter 6. Calculation time increases as we add more CPU greedy algorithms to the IPC pool. This leads us to the first target of the next generation of the framework which is parallelizing the calculation of the DE algorithm where we can setup one master processor that distribute 500 population members to 500 slave processors which in turn they could split the image into multiple segment and distribute it to second level slaves. Then the results are gathered back to the master processor for decision processing. We can also use GPU processing with thousands of processors that could be assigned individually to every pixel such that the calculation on every pixel is done on one GPU processors.

.

References:

[1] K. Price and R. Storn, “Differential evolution,” *Dr Dobbs J.*, vol. 22, no. 4, pp. 18–20, Apr. 1997.

[2] R. Storn and K. Price, *Differential Evolution - A simple and efficient adaptive scheme for global optimization over continuous spaces*. 1995.

[3] R. Storn and K. Price, “Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces,” *J. Glob. Optim.*, vol. 11, no. 4, pp. 341–359, Dec. 1997.

[4] R. Storn, “Differential Evolution Research – Trends and Open Questions.”

[5] R. Storn and K. Price, “Minimizing the real functions of the ICEC’96 contest by differential evolution,” 1996, pp. 842–844.

[6] R. Storn, “On the usage of differential evolution for function optimization,” 1996, pp. 519–523.

[7] J. C. Russ and F. B. Neal, “The image processing handbook 7th edition,” in *The image processing handbook 7th edition*, 7th ed., Taylor & Francis Group, LLC, 20150804, p. 381.

[8] R. C. Gonzales and R. E. Woods, “Digital Image Processing,” in *Digital Image Processing*, Third Edition., 1 vols., Pearson Prentice Hall, pp. 689–690.

[9] “Optic disc detection in color fundus images using ant colony optimization - ProQuest.” [Online]. Available: http://search.proquest.com.uproxy.library.dc-uoit.ca/docview/1314097027?pq-origsite=360link. [Accessed: 07-Apr-2016].

[10] C. Pereira, L. Gonçalves, and M. Ferreira, “Exudate segmentation in fundus images using an ant colony optimization approach,” *Inf. Sci.*, vol. 296, pp. 14–24, Mar. 2015.

[11] X. Wang, Y. Feng, and Z. Feng, “Ant colony optimization with active contour models for image segmentation,” *Kongzhi Lilun Yu YinyongControl Theory Appl.*, vol. 23, no. 4, Aug. 2006.

[12] Y. Han and P. Shi, “An improved ant colony algorithm for fuzzy clustering in image segmentation,” *Neurocomputing*, vol. 70, no. 4–6, pp. 665–671, Jan. 2007.

[13] F. Yuanjing, Y. Li, and K. Liangjun, “Finite grade pheromone ant colony optimization for image segmentation,” *Opto-Electron. Rev.*, vol. 16, no. 2, pp. 163–171, Mar. 2008.

[14] Z. Yu, W. Yu, R. Zou, and S. Yu, “On ACO-Based Fuzzy Clustering for Image Segmentation,” in *Advances in Neural Networks – ISNN 2009*, vol. 5552, W. Yu, H. He, and N. Zhang, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 717–726.

[15] A. Biniaz and A. Abbasi, “Unsupervised ACO: Applying FCM as a supervisor for ACO in medical image segmentation,” *J. Intell. Fuzzy Syst.*, vol. 27, no. 1, pp. 407–417, Jul. 2014.

[16] Y.-C. Liang and Y.-C. Yin, “Optimal multilevel thresholding using a hybrid ant colony system,” *J. Chin. Inst. Ind. Eng.*, vol. 28, no. 1, pp. 20–33, Jan. 2011.

[17] M.-H. Horng, “Multilevel thresholding selection based on the artificial bee colony algorithm for image segmentation,” *Expert Syst. Appl.*, vol. 38, no. 11, pp. 13785–13791, Oct. 2011.

[18] M. Taherdangkoo, “Skull removal in MR images using a modified artificial bee colony optimization algorithm,” *Technol. Health Care*, vol. 22, no. 5, pp. 775–784, Sep. 2014.

[19] J. Li, Y. Zhao, J. Li, and X. Liu, “Artificial Bee Colony Optimizer with Bee-to-Bee Communication and Multipopulation Coevolution for Multilevel Threshold Image Segmentation,” *Math. Probl. Eng.*, vol. 2015, Apr. 2015.

[20] “A Threshold Selection Method from Gray-Level Histograms,” *IEEE Trans. Syst. Man Cybern.*, vol. 9, no. 1, pp. 62–66, Jan. 1979.

[21] R. C. Gonzalez and R. E. Woods, “Otsu,” in *Digital Image Processing*, 3 edition., Upper Saddle River, N.J: Pearson, 2007, pp. 742–747.

[22] L. Ma, K. Hu, Y. Zhu, H. Chen, and M. He, “A Novel Plant Root Foraging Algorithm for Image Segmentation Problems,” *Math. Probl. Eng.*, vol. 2014, pp. 1–16, 2014.

[23] G. G. McNickle, C. C. S. Clair, and J. F. Cahill, “Focusing the metaphor: plant root foraging behaviour,” *Trends Ecol. Evol.*, vol. 24, no. 8, pp. 419–426, Aug. 2009.

[24] Z.-W. Ye, M.-W. Wang, W. Liu, and S.-B. Chen, “Fuzzy entropy based optimal thresholding using bat algorithm,” *Appl. Soft Comput.*, vol. 31, pp. 381–395, Jun. 2015.

[25] V. Rajinikanth and M. S. Couceiro, “Optimal Multilevel Image Threshold Selection Using a Novel Objective Function,” in *Information Systems Design and Intelligent Applications*, vol. 340, J. K. Mandal, S. C. Satapathy, M. Kumar Sanyal, P. P. Sarkar, and A. Mukhopadhyay, Eds. New Delhi: Springer India, 2015, pp. 177–186.

[26] A. Alihodzic and M. Tuba, “Improved Bat Algorithm Applied to Multilevel Image Thresholding,” *Sci. World J.*, vol. 2014, pp. 1–16, 2014.

[27] M. M. S. J. Preetha, L. P. Suresh, and M. J. Bosco, “Cuckoo Search Based Color Image Segmentation Using Seeded Region Growing,” in *Power Electronics and Renewable Energy Systems*, vol. 326, C. Kamalakannan, L. P. Suresh, S. S. Dash, and B. K. Panigrahi, Eds. New Delhi: Springer India, 2015, pp. 1573–1583.

[28] S. Suresh and S. Lal, “An Efficient Cuckoo Search Algorithm based Multilevel Thresholding for Segmentation of Satellite Images Using Different Objective Functions,” *Expert Syst. Appl.*

[29] A. K. Bhandari, V. K. Singh, A. Kumar, and G. K. Singh, “Cuckoo search algorithm and wind driven optimization based study of satellite image segmentation for multilevel thresholding using Kapur’s entropy,” *Expert Syst. Appl.*, vol. 41, no. 7, pp. 3538–3560, Jun. 2014.

[30] A. K. Bhandari, A. Kumar, and G. K. Singh, “Tsallis entropy based multilevel thresholding for colored satellite image segmentation using evolutionary algorithms,” *Expert Syst. Appl.*, vol. 42, no. 22, pp. 8707–8730, Dec. 2015.

[31] S. Agrawal, R. Panda, S. Bhuyan, and B. K. Panigrahi, “Tsallis entropy based optimal multilevel thresholding using cuckoo search algorithm,” *Swarm Evol. Comput.*, vol. 11, pp. 16–30, Aug. 2013.

[32] A. Khan, M. A. Jaffar, and L. Shao, “A modified adaptive differential evolution algorithm for color image segmentation,” *Knowl. Inf. Syst.*, vol. 43, no. 3, pp. 583–597, Mar. 2014.

[33] S. Sarkar, S. Das, and S. S. Chaudhuri, “A multilevel color image thresholding scheme based on minimum cross entropy and differential evolution,” *Pattern Recognit. Lett.*, vol. 54, pp. 27–35, Mar. 2015.

[34] H. V. H. Ayala, F. M. dos Santos, V. C. Mariani, and L. dos S. Coelho, “Image thresholding segmentation based on a novel beta differential evolution approach,” *Expert Syst. Appl.*, vol. 42, no. 4, pp. 2136–2142, Mar. 2015.

[35] S. Sarkar, S. Das, and S. S. Chaudhuri, “Hyper-spectral image segmentation using Rényi entropy based multi-level thresholding aided with differential evolution,” *Expert Syst. Appl.*, vol. 50, pp. 120–129, May 2016.

[36] A. Rényi, “On Measures of Entropy and Information,” presented at the Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of Statistics, 1961.

[37] H. Liu, H. Hu, X. Xu, and E. Song, “Automatic Left Ventricle Segmentation in Cardiac MRI Using Topological Stable-State Thresholding and Region Restricted Dynamic Programming,” *Acad. Radiol.*, vol. 19, no. 6, pp. 723–731, Jun. 2012.

[38] P. Mainali, Qiong Yang, G. Lafruit, L. Van Gool, and R. Lauwereins, “Robust Low Complexity Corner Detector,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 4, pp. 435–445, Apr. 2011.

[39] B. Qi, H. Li, X. Pan, and Y. Kang, “Breast mass segmentation in mammography using improved dynamic programming,” in *2012 International Conference on Information and Automation (ICIA)*, 2012, pp. 215–220.

[40] P. Maduskar, L. Hogeweg, P. A. de Jong, L. Peters-Bax, R. Dawson, H. Ayles, C. I. Sánchez, and B. van Ginneken, “Cavity contour segmentation in chest radiographs using supervised learning and dynamic programming,” *Med. Phys.*, vol. 41, no. 7, p. 71912, Jul. 2014.

[41] L. Hogeweg, C. I. Sánchez, P. A. de Jong, P. Maduskar, and B. van Ginneken, “Clavicle segmentation in chest radiographs,” *Med. Image Anal.*, vol. 16, no. 8, pp. 1490–1502, Dec. 2012.

[42] Pingkun Yan, Sheng Xu, B. Turkbey, and J. Kruecker, “Discrete Deformable Model Guided by Partial Active Shape Model for TRUS Image Segmentation,” *IEEE Trans. Biomed. Eng.*, vol. 57, no. 5, pp. 1158–1166, May 2010.

[43] Y. Zhou, X. Cheng, X. Xu, and E. Song, “Dynamic programming in parallel boundary detection with application to ultrasound intima-media segmentation,” *Med. Image Anal.*, vol. 17, no. 8, pp. 892–906, Dec. 2013.

[44] E. Song, S. Xu, X. Xu, J. Zeng, Y. Lan, S. Zhang, and C.-C. Hung, “Hybrid Segmentation of Mass in Mammograms Using Template Matching and Dynamic Programming,” *Acad. Radiol.*, vol. 17, no. 11, pp. 1414–1424, Nov. 2010.

[45] G. Zahnd, A. Karanasos, G. Soest, E. Regar, W. Niessen, F. Gijsen, and T. Walsum, “Quantification of fibrous cap thickness in intracoronary optical coherence tomography with a contour segmentation method based on dynamic programming,” *Int. J. Comput. Assist. Radiol. Surg.*, vol. 10, no. 9, pp. 1383–1394, Mar. 2015.

[46] X. Qian, Y. Lin, Y. Zhao, J. Wang, J. Liu, and X. Zhuang, “Segmentation of myocardium from cardiac MR images using a novel dynamic programming based segmentation method,” *Med. Phys.*, vol. 42, no. 3, pp. 1424–1435, Mar. 2015.

[47] R. Rocha, A. Campilho, J. Silva, E. Azevedo, and R. Santos, “Segmentation of the carotid intima-media region in B-mode ultrasound images,” *Image Vis. Comput.*, vol. 28, no. 4, pp. 614–625, Apr. 2010.

[48] O. C. Avila-Montes, U. Kurkure, R. Nakazato, D. S. Berman, D. Dey, and I. A. Kakadiaris, “Segmentation of the Thoracic Aorta in Noncontrast Cardiac CT Images,” *IEEE J. Biomed. Health Inform.*, vol. 17, no. 5, pp. 936–949, Sep. 2013.

[49] D. Oliva, E. Cuevas, G. Pajares, D. Zaldivar, and V. Osuna, “A Multilevel Thresholding algorithm using electromagnetism optimization,” *Neurocomputing*, vol. 139, pp. 357–381, Sep. 2014.

[50] D. Oliva, V. Osuna-Enciso, E. Cuevas, G. Pajares, M. Pérez-Cisneros, and D. Zaldívar, “Improving segmentation velocity using an evolutionary method,” *Expert Syst. Appl.*, vol. 42, no. 14, pp. 5874–5886, Aug. 2015.

[51] F. Liu, “The Two Step Mutation Evolutionary Programming Using in Image Segmentation,” *Adv. Mater. Res.*, vol. 433–440, pp. 5459–5462, Jan. 2012.

[52] M. K. Alsmadi, “A HYBRID FIREFLY ALGORITHM WITH FUZZY-C MEAN ALGORITHM FOR MRI BRAIN SEGMENTATION,” *Am. J. Appl. Sci.*, vol. 11, no. 9, pp. 1676–1691, Sep. 2014.

[53] T. Hassanzadeh and H. R. Kanan, “Fuzzy Fa: A Modified Firefly Algorithm,” *Appl. Artif. Intell.*, vol. 28, no. 1, pp. 47–65, Jan. 2014.

[54] K. Chen, Y. Zhou, Z. Zhang, M. Dai, Y. Chao, J. Shi, K. Chen, Y. Zhou, Z. Zhang, M. Dai, Y. Chao, and J. Shi, “Multilevel Image Segmentation Based on an Improved Firefly Algorithm, Multilevel Image Segmentation Based on an Improved Firefly Algorithm,” *Math. Probl. Eng. Math. Probl. Eng.*, vol. 2016, 2016, p. e1578056, Feb. 2016.

[55] M. H. Horng and T. W. Jiang, “Multilevel Image Thresholding Selection Based on the Firefly Algorithm,” in *2010 7th International Conference on Ubiquitous Intelligence Computing and 7th International Conference on Autonomic Trusted Computing (UIC/ATC)*, 2010, pp. 58–63.

[56] M.-H. Horng and R.-J. Liou, “Multilevel minimum cross entropy threshold selection based on the firefly algorithm,” *Expert Syst. Appl.*, vol. 38, no. 12, pp. 14805–14811, Nov. 2011.

[57] A. Ladgham, F. Hamdaoui, A. Sakly, and A. Mtibaa, “Fast MR brain image segmentation based on modified Shuffled Frog Leaping Algorithm,” *Signal Image Video Process.*, vol. 9, no. 5, pp. 1113–1120, Jul. 2015.

[58] N. Wang, X. Li, and X. Chen, “Fast three-dimensional Otsu thresholding with shuffled frog-leaping algorithm,” *Pattern Recognit. Lett.*, vol. 31, no. 13, pp. 1809–1815, Oct. 2010.

[59] M. H. Horng, “Multilevel image threshold selection based on the shuffled frog-leaping algorithm,” *ResearchGate*, vol. 5, no. 9, pp. 599–605, Jan. 2013.

[60] M. Horng, “Multilevel Minimum Cross Entropy Threshold Selection with Shuffled Frog-leaping Algorithm,” *Int. J. Adv. Inf. Sci. Serv. Sci.*, vol. 4, no. 11, pp. 168–176, Jun. 2012.

[61] O. Khayat, F. N. Rahatabad, M. Siahi, and B. Azadbakht, “An evolutionary-based entropic image thresholding approach for nano-scale light microscopic image segmentation,” *J. Intell. Fuzzy Syst.*, vol. 27, no. 6, pp. 2959–2967, Dec. 2014.

[62] A. G. Shanbhag, “Utilization of Information Measure as a Means of Image Thresholding,” *CVGIP Graph. Models Image Process.*, vol. 56, no. 5, pp. 414–419, Sep. 1994.

[63] F. Xie and A. C. Bovik, “Automatic segmentation of dermoscopy images using self-generating neural networks seeded by genetic algorithm,” *Pattern Recognit.*, vol. 46, no. 3, pp. 1012–1019, Mar. 2013.

[64] P. Wang, X. Hu, Y. Li, Q. Liu, and X. Zhu, “Automatic cell nuclei segmentation and classification of breast cancer histopathology images,” *Signal Process.*, vol. 122, pp. 1–13, May 2016.

[65] R. Usamentiaga, “Automatic tuning for the segmentation of infrared images considering uncertain ground truth,” *J. Electron. Imaging*, vol. 18, no. 1, p. 13001, Jan. 2009.

[66] A. Khan, J. Ullah, M. A. Jaffar, and T.-S. Choi, “Color image segmentation: a novel spatial fuzzy genetic algorithm,” *Signal Image Video Process.*, vol. 8, no. 7, pp. 1233–1243, Oct. 2014.

[67] J. S. Lim, “Two-Dimensional Signal and Image Processing,” 1 edition., Englewood Cliffs, N.J: Prentice Hall PTR, 1989, pp. 469–476.

[68] R. N. Czerwinski, D. L. Jones, and W. D. O’Brien, “Detection of lines and boundaries in speckle images-application to medical ultrasound,” *IEEE Trans. Med. Imaging*, vol. 18, no. 2, pp. 126–136, Feb. 1999.

[69] A. Polesel, G. Ramponi, and V. J. Mathews, “Image enhancement via adaptive unsharp masking,” *IEEE Trans. Image Process.*, vol. 9, no. 3, pp. 505–510, Mar. 2000.

[70] R. C. Gonzalez and R. E. Woods, “Sharpening Spatial Filters,” in *Digital Image Processing*, 3 edition., Upper Saddle River, N.J: Pearson, 2007, pp. 162-165-289.

[71] R. C. Gonzalez and R. E. Woods, “Minimum Mean Square Error (Weiner) Filtering,” in *Digital Image Processing*, 3 edition., Upper Saddle River, N.J: Pearson, 2007, pp. 352–357.

[72] J. C. Russ and F. B. Neal, “The Image Processing Handbook, Seventh Edition,” in *The Image Processing Handbook, Seventh Edition*, 7 edition., Boca Raton: CRC Press, 2015, pp. 358–362.

[73] R. C. Gonzalez and R. E. Woods, “Smoothing Spatial Filters,” in *Digital Image Processing*, 3 edition., Upper Saddle River, N.J: Pearson, 2007, p. 152.

[74] R. A. Haddad and A. N. Akansu, “A class of fast Gaussian binomial filters for speech and image processing,” *IEEE Trans. Signal Process.*, vol. 39, no. 3, pp. 723–727, Mar. 1991.

[75] J. C. Russ and F. B. Neal, “Gaussian smoothing,” in *The Image Processing Handbook, Seventh Edition*, 7 edition., Boca Raton: CRC Press, 2015, pp. 187, 340.

[76] D. Marr and E. Hildreth, “Theory of Edge Detection,” *Proc. R. Soc. Lond. B Biol. Sci.*, vol. 207, no. 1167, pp. 187–217, 1980.

[77] R. C. Gonzalez and R. E. Woods, “Point, Line, and Edge Detection,” in *Digital Image Processing*, 3 edition., Upper Saddle River, N.J: Pearson, 2007, p. 715,789.

[78] J. C. Russ and F. B. Neal, “Edge Detection,” in *The Image Processing Handbook, Seventh Edition*, 7 edition., Boca Raton: CRC Press, 2015, pp. 271, 286, 614–615.

[79] R. C. Gonzalez and R. E. Woods, “Sharpening Spatial Filters,” in *Digital Image Processing*, 3 edition., Upper Saddle River, N.J: Pearson, 2007, pp. 160–162.

[80] J. C. Russ and F. B. Neal, “Contrast enhancement,” in *The Image Processing Handbook, Seventh Edition*, 7 edition., Boca Raton: CRC Press, 2015, pp. 261–268, 276, 284, 820.

[81] S. Cui, Y. Wang, X. Qian, and Z. Deng, “Image Processing Techniques in Shockwave Detection and Modeling,” *J. Signal Inf. Process.*, vol. 4, no. 3, pp. 109–113, 2013.

[82] B. S. Lipkin, Ed., “Object enhancement and extraction,” in *Picture Processing and Psychopictorics*, Academic Press, 1970, pp. 75–149.

[83] L. S. Davis, “A survey of edge detection techniques,” *Comput. Graph. Image Process.*, vol. 4, no. 3, pp. 248–270, Sep. 1975.

[84] I. Sobel, “History and Definition of the Sobel Operator.” .

[85] R. van den Boomgaard and R. van Balen, “Methods for fast morphological image transforms using bitmapped binary images,” *CVGIP Graph. Models Image Process.*, vol. 54, no. 3, pp. 252–258, May 1992.

[86] R. C. Gonzalez and R. E. Woods, “Errosion,” in *Digital Image Processing*, 3 edition., Upper Saddle River, N.J: Pearson, 2007, pp. 631–633.

[87] R. Adams, “Radial Decomposition of Disks and Spheres,” *CVGIP Graph. Models Image Process.*, vol. 55, no. 5, pp. 325–332, Sep. 1993.

[88] R. C. Gonzalez and R. E. Woods, “Dilation,” in *Digital Image Processing*, 3 edition., Upper Saddle River, N.J: Pearson, 2007, pp. 633–635.

[89] R. C. Gonzalez and R. E. Woods, “Opening and Closing,” in *Digital Image Processing*, 3 edition., Upper Saddle River, N.J: Pearson, 2007, pp. 635–639.

[90] R. Jones and P. Soille, “Periodic lines: Definition, cascades, and application to granulometries,” *Pattern Recognit. Lett.*, vol. 17, no. 10, pp. 1057–1063, Sep. 1996.

[91] J. Kittler and J. Illingworth, “Minimum error thresholding,” *Pattern Recognit.*, vol. 19, no. 1, pp. 41–47, Jan. 1986.

[92] J. R. Parker, in *Algorithms for Image Processing and Computer Vision*, 2 edition., New York: Wiley, 2010, pp. 148–149.

[93] P. K. Sahoo, S. Soltani, and A. K. C. Wong, “A survey of thresholding techniques,” *Comput. Vis. Graph. Image Process.*, vol. 41, no. 2, pp. 233–260, Feb. 1988.

[94] A. D. Brink and N. E. Pendock, “Minimum cross-entropy threshold selection,” *Pattern Recognit.*, vol. 29, no. 1, pp. 179–188, Jan. 1996.

[95] G. W. Zack, W. E. Rogers, and S. A. Latt, “Automatic measurement of sister chromatid exchange frequency.,” *J. Histochem. Cytochem.*, vol. 25, no. 7, pp. 741–753, Jul. 1977.

[96] R. C. Gonzalez and R. E. Woods, “Hole Filling,” in *Digital Image Processing*, 3 edition., Upper Saddle River, N.J: Pearson, 2007, pp. 643–645.

[97] D. P. Soille, “Geodesic Transformations,” in *Morphological Image Analysis*, Springer Berlin Heidelberg, 2004, pp. 183–218.

[98] D. P. Soille, “Erosion and Dilation,” in *Morphological Image Analysis*, Springer Berlin Heidelberg, 2004, pp. 63–103.

[99] D. P. Soille, “Opening and Closing,” in *Morphological Image Analysis*, Springer Berlin Heidelberg, 2004, pp. 105–137.

[100] R. C. Gonzalez and R. E. Woods, “Intensity Transformation and Spatial Filtering,” in *Digital Image Processing*, 3 edition., Upper Saddle River, N.J: Pearson, 2007, pp. 115–116.

[101] J. C. Russ and F. B. Neal, “Image Enhancement in the Spatial Domain,” in *The Image Processing Handbook, Seventh Edition*, 7 edition., Boca Raton: CRC Press, 2015, pp. 253–255.

[102] R. C. Gonzalez and R. E. Woods, “Histogram Processing,” in *Digital Image Processing*, 3 edition., Upper Saddle River, N.J: Pearson, 2007, pp. 162-165-289.

[103] P. Heckbert, Ed., “Contrast Limited Adaptive Histogram Equalization,” in *Graphics Gems IV*, 1 edition., Boston: Morgan Kaufmann, 1994, pp. 474–485.

[104] S. M. Pizer, E. P. Amburn, J. D. Austin, R. Cromartie, A. Geselowitz, T. Greer, B. ter Haar Romeny, J. B. Zimmerman, and K. Zuiderveld, “Adaptive histogram equalization and its variations,” *Comput. Vis. Graph. Image Process.*, vol. 39, no. 3, pp. 355–368, Sep. 1987.

[105] “Fuzzy Image Enhancment Chapter 5_tizhoosh2000.pdf.” .

[106] H. R. Tizhoosh, G. Krell, and B. Michaelis, “λ-enhancement: contrast adaptation based on optimization of image fuzziness,” 1998, vol. 2, pp. 1548–1553.

[107] “JSRT Database | Japanese Society of Radiological Technology.” [Online]. Available: http://www.jsrt.or.jp/jsrt-db/eng.php. [Accessed: 04-May-2016].

Appendices

8.1 Appendix A

Image Name	Sensitivity	Precession	Overlap
JPCLN001.tif	87.84%	95.76%	84.55%
JPCLN002.tif	90.44%	42.67%	40.83%
JPCLN003.tif	84.55%	87.13%	75.17%
JPCLN004.tif	89.48%	86.43%	78.46%
JPCLN005.tif	83.52%	79.86%	68.99%
JPCLN006.tif	82.71%	84.47%	71.79%
JPCLN007.tif	86.29%	91.37%	79.79%
JPCLN008.tif	78.79%	91.34%	73.31%
JPCLN009.tif	80.71%	39.47%	36.07%
JPCLN010.tif	65.31%	87.02%	59.51%
JPCLN011.tif	89.07%	71.99%	66.15%
JPCLN012.tif	84.98%	80.16%	70.21%
JPCLN013.tif	89.22%	89.35%	80.64%
JPCLN014.tif	85.69%	93.63%	80.97%
JPCLN015.tif	84.15%	81.81%	70.89%
JPCLN016.tif	88.47%	47.58%	44.80%
JPCLN017.tif	81.93%	87.36%	73.25%
JPCLN018.tif	88.43%	75.55%	68.75%
JPCLN019.tif	84.41%	91.50%	78.27%
JPCLN020.tif	74.73%	86.23%	66.76%
JPCLN021.tif	89.27%	87.85%	79.46%
JPCLN022.tif	84.37%	90.00%	77.14%
JPCLN023.tif	86.11%	87.07%	76.35%
JPCLN024.tif	72.52%	91.03%	67.68%
JPCLN025.tif	77.09%	82.88%	66.50%
JPCLN026.tif	88.50%	92.64%	82.68%
JPCLN027.tif	80.44%	88.77%	73.01%
JPCLN028.tif	85.88%	89.37%	77.92%
JPCLN029.tif	81.21%	86.65%	72.17%
JPCLN030.tif	77.46%	84.01%	67.51%
JPCLN031.tif	78.99%	91.11%	73.34%
JPCLN032.tif	88.46%	91.95%	82.10%
JPCLN033.tif	86.52%	51.22%	47.44%
JPCLN034.tif	90.02%	63.81%	59.59%
JPCLN035.tif	87.25%	88.71%	78.53%
JPCLN036.tif	91.98%	67.69%	63.91%
JPCLN037.tif	69.73%	86.03%	62.64%
JPCLN038.tif	89.25%	86.29%	78.17%
JPCLN039.tif	63.33%	48.38%	37.79%
JPCLN040.tif	87.28%	69.29%	62.94%

JPCLN041.tif	86.80%	71.61%	64.57%
JPCLN042.tif	80.27%	78.61%	65.88%
JPCLN043.tif	76.53%	86.31%	68.24%
JPCLN044.tif	78.70%	92.15%	73.75%
JPCLN045.tif	84.73%	94.37%	80.65%
JPCLN046.tif	91.00%	76.11%	70.78%
JPCLN047.tif	85.68%	65.67%	59.18%
JPCLN048.tif	85.33%	59.92%	54.32%
JPCLN049.tif	84.19%	89.85%	76.88%
JPCLN050.tif	77.87%	91.80%	72.80%
JPCLN051.tif	82.42%	93.64%	78.05%
JPCLN052.tif	83.69%	89.50%	76.20%
JPCLN053.tif	85.82%	93.07%	80.67%
JPCLN054.tif	85.75%	91.99%	79.79%
JPCLN055.tif	88.04%	86.50%	77.40%
JPCLN056.tif	88.02%	92.66%	82.28%
JPCLN057.tif	83.86%	89.98%	76.70%
JPCLN058.tif	91.60%	79.65%	74.23%
JPCLN059.tif	87.49%	83.52%	74.61%
JPCLN060.tif	81.62%	89.64%	74.58%
JPCLN061.tif	81.63%	86.99%	72.75%
JPCLN062.tif	80.01%	90.49%	73.80%
JPCLN063.tif	86.57%	90.06%	79.02%
JPCLN064.tif	89.23%	51.48%	48.47%
JPCLN065.tif	87.54%	88.27%	78.42%
JPCLN066.tif	85.79%	80.25%	70.83%
JPCLN067.tif	87.92%	82.59%	74.18%
JPCLN068.tif	86.59%	91.82%	80.39%
JPCLN069.tif	82.42%	88.79%	74.66%
JPCLN070.tif	84.19%	67.26%	59.72%
JPCLN071.tif	79.42%	87.58%	71.38%
JPCLN072.tif	83.73%	80.87%	69.89%
JPCLN073.tif	86.46%	90.71%	79.42%
JPCLN074.tif	86.66%	69.88%	63.09%
JPCLN075.tif	67.98%	94.33%	65.31%
JPCLN076.tif	86.90%	89.78%	79.08%
JPCLN077.tif	84.68%	52.59%	48.02%
JPCLN078.tif	83.91%	90.54%	77.15%
JPCLN079.tif	84.89%	90.84%	78.19%
JPCLN080.tif	84.30%	88.08%	75.66%
JPCLN081.tif	79.59%	47.91%	42.67%
JPCLN082.tif	86.60%	53.73%	49.60%

JPCLN083.tif	80.93%	92.01%	75.61%
JPCLN084.tif	82.05%	93.58%	77.68%
JPCLN085.tif	86.68%	81.09%	72.10%
JPCLN086.tif	81.98%	76.97%	65.83%
JPCLN087.tif	85.49%	89.60%	77.77%
JPCLN088.tif	80.81%	85.63%	71.16%
JPCLN089.tif	86.29%	87.75%	77.01%
JPCLN090.tif	83.29%	70.99%	62.14%
JPCLN091.tif	85.43%	90.11%	78.11%
JPCLN092.tif	89.91%	78.23%	71.91%
JPCLN093.tif	79.47%	95.71%	76.74%
JPCLN094.tif	86.20%	94.45%	82.04%
JPCLN095.tif	74.25%	90.23%	68.72%
JPCLN096.tif	82.23%	90.30%	75.55%
JPCLN097.tif	88.59%	81.19%	73.50%
JPCLN098.tif	85.21%	88.78%	76.93%
JPCLN099.tif	83.11%	87.89%	74.57%
JPCLN100.tif	83.45%	90.54%	76.76%
JPCLN101.tif	79.97%	79.66%	66.41%
JPCLN102.tif	84.93%	92.28%	79.29%
JPCLN103.tif	86.27%	68.88%	62.08%
JPCLN104.tif	86.76%	92.26%	80.87%
JPCLN105.tif	82.08%	90.55%	75.60%
JPCLN106.tif	86.02%	85.87%	75.35%
JPCLN107.tif	88.45%	83.29%	75.12%
JPCLN108.tif	72.71%	93.24%	69.07%
JPCLN109.tif	87.01%	90.18%	79.48%
JPCLN110.tif	82.92%	92.65%	77.80%
JPCLN111.tif	86.06%	92.40%	80.36%
JPCLN112.tif	89.82%	58.65%	54.99%
JPCLN113.tif	81.99%	89.57%	74.85%
JPCLN114.tif	83.67%	93.39%	78.99%
JPCLN115.tif	82.87%	87.24%	73.91%
JPCLN116.tif	82.46%	91.37%	76.51%
JPCLN117.tif	87.50%	78.82%	70.84%
JPCLN118.tif	86.21%	81.04%	71.74%
JPCLN119.tif	81.77%	86.95%	72.83%
JPCLN120.tif	86.73%	95.62%	83.42%
JPCLN121.tif	85.27%	83.02%	72.61%
JPCLN122.tif	80.91%	50.70%	45.28%
JPCLN123.tif	83.79%	90.63%	77.11%
JPCLN124.tif	84.39%	91.41%	78.20%

JPCLN125.tif	82.56%	94.00%	78.42%
JPCLN126.tif	89.75%	84.12%	76.75%
JPCLN127.tif	71.19%	88.99%	65.43%
JPCLN128.tif	87.66%	95.40%	84.10%
JPCLN129.tif	82.11%	90.68%	75.72%
JPCLN130.tif	83.19%	82.12%	70.43%
JPCLN131.tif	81.89%	88.81%	74.24%
JPCLN132.tif	84.53%	88.57%	76.21%
JPCLN133.tif	83.20%	89.06%	75.49%
JPCLN134.tif	83.96%	85.23%	73.30%
JPCLN135.tif	90.04%	63.26%	59.13%
JPCLN136.tif	82.81%	49.28%	44.70%
JPCLN137.tif	80.72%	89.55%	73.77%
JPCLN138.tif	85.41%	88.97%	77.23%
JPCLN139.tif	87.25%	88.61%	78.45%
JPCLN140.tif	85.32%	89.94%	77.89%
JPCLN141.tif	81.56%	75.33%	64.37%
JPCLN142.tif	85.98%	87.62%	76.67%
JPCLN143.tif	89.21%	77.28%	70.67%
JPCLN144.tif	83.38%	91.68%	77.51%
JPCLN145.tif	85.23%	92.41%	79.65%
JPCLN146.tif	82.03%	92.15%	76.67%
JPCLN147.tif	84.18%	87.39%	75.07%
JPCLN148.tif	79.23%	88.05%	71.54%
JPCLN149.tif	85.69%	92.10%	79.82%
JPCLN150.tif	86.37%	91.40%	79.87%
JPCLN151.tif	83.78%	90.59%	77.08%
JPCLN152.tif	84.99%	92.16%	79.25%
JPCLN153.tif	79.58%	85.45%	70.08%
JPCLN154.tif	85.05%	81.82%	71.53%
JPCNN001.tif	85.36%	86.22%	75.11%
JPCNN002.tif	81.42%	87.96%	73.25%
JPCNN003.tif	72.15%	91.56%	67.65%
JPCNN004.tif	83.45%	91.30%	77.30%
JPCNN005.tif	86.86%	88.34%	77.93%
JPCNN006.tif	90.07%	92.84%	84.22%
JPCNN007.tif	88.90%	62.53%	58.00%
JPCNN008.tif	81.24%	73.30%	62.69%
JPCNN009.tif	84.23%	91.47%	78.10%
JPCNN010.tif	88.55%	79.25%	71.88%
JPCNN011.tif	87.43%	88.42%	78.44%
JPCNN012.tif	91.47%	85.48%	79.17%

JPCNN013.tif	83.83%	89.00%	75.95%
JPCNN014.tif	84.54%	87.88%	75.71%
JPCNN015.tif	79.21%	83.18%	68.27%
JPCNN016.tif	86.76%	80.77%	71.91%
JPCNN017.tif	87.63%	90.89%	80.55%
JPCNN018.tif	82.76%	89.71%	75.58%
JPCNN019.tif	83.60%	92.03%	77.96%
JPCNN020.tif	85.05%	91.42%	78.76%
JPCNN021.tif	85.02%	88.18%	76.32%
JPCNN022.tif	88.52%	82.03%	74.14%
JPCNN023.tif	76.66%	84.13%	66.98%
JPCNN024.tif	70.20%	84.92%	62.42%
JPCNN025.tif	84.79%	87.26%	75.45%
JPCNN026.tif	87.29%	91.40%	80.66%
JPCNN027.tif	83.68%	81.57%	70.38%
JPCNN028.tif	85.00%	89.16%	77.04%
JPCNN029.tif	84.85%	88.00%	76.05%
JPCNN030.tif	86.92%	82.57%	73.45%
JPCNN031.tif	87.28%	79.23%	71.02%
JPCNN032.tif	81.47%	77.84%	66.13%
JPCNN033.tif	87.29%	87.47%	77.59%
JPCNN034.tif	82.47%	79.49%	68.00%
JPCNN035.tif	84.75%	88.74%	76.52%
JPCNN036.tif	88.11%	90.36%	80.54%
JPCNN037.tif	86.49%	95.25%	82.91%
JPCNN038.tif	75.26%	93.02%	71.24%
JPCNN039.tif	77.74%	91.16%	72.29%
JPCNN040.tif	81.29%	81.72%	68.79%
JPCNN041.tif	88.67%	86.76%	78.10%
JPCNN042.tif	68.37%	93.30%	65.16%
JPCNN043.tif	78.13%	88.07%	70.65%
JPCNN044.tif	85.14%	68.24%	60.98%
JPCNN045.tif	86.36%	82.60%	73.07%
JPCNN046.tif	81.76%	77.08%	65.77%
JPCNN047.tif	84.08%	75.85%	66.32%
JPCNN048.tif	84.98%	84.73%	73.69%
JPCNN049.tif	83.60%	71.23%	62.49%
JPCNN050.tif	84.95%	77.63%	68.24%
JPCNN051.tif	75.41%	86.96%	67.75%
JPCNN052.tif	79.90%	85.62%	70.45%
JPCNN053.tif	82.45%	88.95%	74.79%
JPCNN054.tif	84.26%	86.87%	74.74%

JPCNN055.tif	88.23%	74.47%	67.74%
JPCNN056.tif	84.55%	81.16%	70.68%
JPCNN057.tif	85.60%	92.34%	79.92%
JPCNN058.tif	82.41%	89.45%	75.11%
JPCNN059.tif	85.52%	88.20%	76.74%
JPCNN060.tif	81.79%	86.91%	72.82%
JPCNN061.tif	87.74%	89.69%	79.69%
JPCNN062.tif	83.72%	85.82%	73.55%
JPCNN063.tif	82.30%	91.23%	76.27%
JPCNN064.tif	87.08%	57.01%	52.56%
JPCNN065.tif	84.50%	83.18%	72.18%
JPCNN066.tif	90.24%	77.90%	71.84%
JPCNN067.tif	81.95%	87.28%	73.21%
JPCNN068.tif	85.91%	87.52%	76.53%
JPCNN069.tif	84.15%	87.79%	75.33%
JPCNN070.tif	80.91%	86.76%	72.01%
JPCNN071.tif	79.47%	86.94%	70.99%
JPCNN072.tif	83.02%	81.33%	69.73%
JPCNN073.tif	85.71%	87.09%	76.05%
JPCNN074.tif	82.94%	82.69%	70.67%
JPCNN075.tif	87.03%	85.38%	75.75%
JPCNN076.tif	85.94%	80.09%	70.81%
JPCNN077.tif	82.91%	77.60%	66.90%
JPCNN078.tif	80.55%	88.97%	73.24%
JPCNN079.tif	81.29%	87.04%	72.51%
JPCNN080.tif	83.36%	85.48%	73.02%
JPCNN081.tif	85.80%	92.87%	80.50%
JPCNN082.tif	83.09%	91.92%	77.43%
JPCNN083.tif	84.01%	89.17%	76.23%
JPCNN084.tif	84.18%	92.18%	78.57%
JPCNN085.tif	82.00%	88.07%	73.80%
JPCNN086.tif	84.13%	86.25%	74.18%
JPCNN087.tif	86.36%	84.81%	74.79%
JPCNN088.tif	85.63%	86.49%	75.53%
JPCNN089.tif	80.91%	92.65%	76.03%
JPCNN090.tif	87.34%	88.12%	78.14%
JPCNN091.tif	87.74%	69.30%	63.18%
JPCNN092.tif	82.91%	80.16%	68.80%
JPCNN093.tif	83.54%	86.59%	73.97%
Average	83.9%	84%	72%
Standard Deviation	0.04	0.11	0.084

8.2 Appendix B

Image Name	Sensitivity	Precession	Overlap
1.tif	4559	66	100
10.tif	1686	40	100
11.tif	177	90	98
12.tif	6726	73	48
13.tif	16099	3	100
14.tif	18038	31	100
15.tif	16597	52	100
16.tif	11564	38	99
17.tif	22710	14	100
18.tif	3695	82	62
19.tif	5805	32	99
2.tif	1537	99	86
20.tif	11619	4	100
21.tif	3160	65	94
22.tif	13151	90	75
23.tif	24845	66	98
24.tif	653	83	100
25.tif	2076	58	90
26.tif	1070	97	84
27.tif	1168	92	91
28.tif	5285	72	100
29.tif	23552	98	42
3.tif	2233	98	88
30.tif	74394	96	20
4.tif	533	86	73
5.tif	1033	55	100
6.tif	7321	48	100
7.tif	4226	70	100
8.tif	10588	69	62
9.tif	19617	93	15
Average	65.33%	84.13%	52.37%
Standard Dev.	28.08	23.95	24.63

8.3 Appendix C

Image Name	Sensitivity	Precession	Overlap
JPCLN001.tif	70	97	68
JPCLN002.tif	71	96	69
JPCLN003.tif	100	10	10
JPCLN004.tif	77	97	75
JPCLN005.tif	100	9	9
JPCLN006.tif	75	98	74
JPCLN007.tif	100	14	14
JPCLN008.tif	68	99	68
JPCLN009.tif	100	13	13
JPCLN010.tif	100	10	10
JPCLN011.tif	100	14	14
JPCLN012.tif	100	13	13
JPCLN013.tif	100	12	12
JPCLN014.tif	70	99	70
JPCLN015.tif	100	13	13
JPCLN016.tif	80	69	59
JPCLN017.tif	100	12	12
JPCLN018.tif	70	98	69
JPCLN019.tif	75	97	74
JPCLN020.tif	71	92	67
JPCLN021.tif	78	97	76
JPCLN022.tif	71	97	70
JPCLN023.tif	64	92	60
JPCLN024.tif	77	96	74
JPCLN025.tif	100	9	9
JPCLN026.tif	78	96	76
JPCLN027.tif	71	97	70
JPCLN028.tif	77	97	75
JPCLN029.tif	100	9	9
JPCLN030.tif	66	94	64
JPCLN031.tif	100	13	13
JPCLN032.tif	83	96	80
JPCLN033.tif	69	99	68
JPCLN034.tif	90	66	62
JPCLN035.tif	76	98	74
JPCLN036.tif	100	15	15
JPCLN037.tif	100	11	11
JPCLN038.tif	91	95	87
JPCLN039.tif	55	99	55
JPCLN040.tif	100	16	16

JPCLN041.tif	100	14	14
JPCLN042.tif	100	9	9
JPCLN043.tif	100	12	12
JPCLN044.tif	100	12	12
JPCLN045.tif	79	98	77
JPCLN046.tif	100	14	14
JPCLN047.tif	73	97	71
JPCLN048.tif	100	7	7
JPCLN049.tif	100	14	14
JPCLN050.tif	72	95	69
JPCLN051.tif	100	13	13
JPCLN052.tif	68	96	66
JPCLN053.tif	73	98	72
JPCLN054.tif	100	12	12
JPCLN055.tif	100	12	12
JPCLN056.tif	81	98	80
JPCLN057.tif	72	97	70
JPCLN058.tif	77	99	77
JPCLN059.tif	100	11	11
JPCLN060.tif	100	12	12
JPCLN061.tif	100	13	13
JPCLN062.tif	69	96	67
JPCLN063.tif	100	17	17
JPCLN064.tif	73	96	71
JPCLN065.tif	100	13	13
JPCLN066.tif	65	99	64
JPCLN067.tif	100	11	11
JPCLN068.tif	78	97	76
JPCLN069.tif	77	94	74
JPCLN070.tif	78	93	74
JPCLN071.tif	66	97	65
JPCLN072.tif	100	11	11
JPCLN073.tif	83	95	80
JPCLN074.tif	100	16	16
JPCLN075.tif	67	98	67
JPCLN076.tif	78	96	76
JPCLN077.tif	78	57	49
JPCLN078.tif	83	94	79
JPCLN079.tif	71	99	71
JPCLN080.tif	82	95	78
JPCLN081.tif	58	97	57
JPCLN082.tif	66	100	66

JPCLN083.tif	65	97	64
JPCLN084.tif	75	99	74
JPCLN085.tif	100	10	10
JPCLN086.tif	81	97	79
JPCLN087.tif	83	97	81
JPCLN088.tif	65	99	65
JPCLN089.tif	100	21	21
JPCLN090.tif	76	99	75
JPCLN091.tif	79	97	77
JPCLN092.tif	85	95	82
JPCLN093.tif	66	99	66
JPCLN094.tif	87	98	85
JPCLN095.tif	100	8	8
JPCLN096.tif	100	13	13
JPCLN097.tif	100	16	16
JPCLN098.tif	82	96	79
JPCLN099.tif	100	10	10
JPCLN100.tif	100	11	11
JPCLN101.tif	83	89	76
JPCLN102.tif	75	98	74
JPCLN103.tif	63	100	62
JPCLN104.tif	86	96	83
JPCLN105.tif	75	95	72
JPCLN106.tif	100	14	14
JPCLN107.tif	83	98	82
JPCLN108.tif	65	95	63
JPCLN109.tif	75	98	74
JPCLN110.tif	75	97	74
JPCLN111.tif	82	95	79
JPCLN112.tif	72	98	71
JPCLN113.tif	79	95	76
JPCLN114.tif	64	100	64
JPCLN115.tif	77	94	73
JPCLN116.tif	100	11	11
JPCLN117.tif	100	10	10
JPCLN118.tif	81	95	78
JPCLN119.tif	100	10	10
JPCLN120.tif	74	100	74
JPCLN121.tif	82	95	79
JPCLN122.tif	54	100	54
JPCLN123.tif	100	13	13
JPCLN124.tif	100	14	14

JPCLN125.tif	55	98	55
JPCLN126.tif	70	99	69
JPCLN127.tif	100	12	12
JPCLN128.tif	61	100	61
JPCLN129.tif	64	99	64
JPCLN130.tif	80	91	74
JPCLN131.tif	79	95	76
JPCLN132.tif	100	11	11
JPCLN133.tif	100	15	15
JPCLN134.tif	84	90	76
JPCLN135.tif	100	13	13
JPCLN136.tif	100	11	11
JPCLN137.tif	100	8	8
JPCLN138.tif	79	97	77
JPCLN139.tif	83	95	79
JPCLN140.tif	80	100	80
JPCLN141.tif	100	15	15
JPCLN142.tif	70	97	68
JPCLN143.tif	78	94	75
JPCLN144.tif	80	97	78
JPCLN145.tif	75	97	73
JPCLN146.tif	76	95	73
JPCLN147.tif	85	91	78
JPCLN148.tif	71	94	68
JPCLN149.tif	100	12	12
JPCLN150.tif	79	98	78
JPCLN151.tif	100	11	11
JPCLN152.tif	71	98	70
JPCLN153.tif	73	96	71
JPCLN154.tif	100	11	11
JPCNN001.tif	79	98	78
JPCNN002.tif	76	87	68
JPCNN003.tif	100	10	10
JPCNN004.tif	83	94	79
JPCNN005.tif	82	97	80
JPCNN006.tif	100	14	14
JPCNN007.tif	100	12	12
JPCNN008.tif	100	14	14
JPCNN009.tif	78	97	76
JPCNN010.tif	83	98	81
JPCNN011.tif	100	11	11
JPCNN012.tif	86	96	83

JPCNN013.tif	100	13	13
JPCNN014.tif	100	13	13
JPCNN015.tif	100	11	11
JPCNN016.tif	85	96	82
JPCNN017.tif	64	99	64
JPCNN018.tif	100	15	15
JPCNN019.tif	74	98	74
JPCNN020.tif	73	95	70
JPCNN021.tif	72	98	71
JPCNN022.tif	87	91	80
JPCNN023.tif	83	95	80
JPCNN024.tif	100	9	9
JPCNN025.tif	75	93	71
JPCNN026.tif	100	13	13
JPCNN027.tif	100	10	10
JPCNN028.tif	73	96	71
JPCNN029.tif	100	8	8
JPCNN030.tif	100	12	12
JPCNN031.tif	100	17	17
JPCNN032.tif	100	16	16
JPCNN033.tif	79	93	75
JPCNN034.tif	100	16	16
JPCNN035.tif	100	15	15
JPCNN036.tif	79	94	76
JPCNN037.tif	86	95	83
JPCNN038.tif	75	97	73
JPCNN039.tif	100	14	14
JPCNN040.tif	85	87	76
JPCNN041.tif	82	91	75
JPCNN042.tif	100	15	15
JPCNN043.tif	100	10	10
JPCNN044.tif	100	15	15
JPCNN045.tif	100	19	19
JPCNN046.tif	100	16	16
JPCNN047.tif	100	13	13
JPCNN048.tif	80	84	69
JPCNN049.tif	100	15	15
JPCNN050.tif	79	95	76
JPCNN051.tif	100	13	13
JPCNN052.tif	72	95	69
JPCNN053.tif	91	88	81
JPCNN054.tif	83	97	80

JPCNN055.tif	89	91	81
JPCNN056.tif	74	90	68
JPCNN057.tif	72	98	71
JPCNN058.tif	71	99	70
JPCNN059.tif	100	16	16
JPCNN060.tif	100	14	14
JPCNN061.tif	100	18	18
JPCNN062.tif	100	9	9
JPCNN063.tif	84	95	81
JPCNN064.tif	71	99	70
JPCNN065.tif	100	12	12
JPCNN066.tif	100	10	10
JPCNN067.tif	100	11	11
JPCNN068.tif	82	95	79
JPCNN069.tif	77	97	75
JPCNN070.tif	100	12	12
JPCNN071.tif	68	97	66
JPCNN072.tif	100	11	11
JPCNN073.tif	77	94	74
JPCNN074.tif	100	14	14
JPCNN075.tif	74	94	71
JPCNN076.tif	78	97	76
JPCNN077.tif	100	11	11
JPCNN078.tif	100	12	12
JPCNN079.tif	90	88	80
JPCNN080.tif	100	8	8
JPCNN081.tif	100	18	18
JPCNN082.tif	100	13	13
JPCNN083.tif	100	11	11
JPCNN084.tif	67	96	65
JPCNN085.tif	67	98	66
JPCNN086.tif	88	93	82
JPCNN087.tif	100	9	9
JPCNN088.tif	100	11	11
JPCNN089.tif	73	96	71
JPCNN090.tif	100	12	12
JPCNN091.tif	74	99	73
JPCNN092.tif	100	8	8
JPCNN093.tif	100	12	12
Average	86.00 %	60.09 %	47.06 %
Standard Dev.	13.30	41.21	30.25

8.4 Appendix D

Image Name	Sensitivity	Precession	Overlap
1.tif	3	100	3
10.tif	16	100	16
11.tif	30	100	30
12.tif	7	100	7
13.tif	3	100	3
14.tif	2	100	2
15.tif	1	100	1
16.tif	2	100	2
17.tif	2	100	2
18.tif	8	100	8
19.tif	5	100	5
2.tif	5	100	5
20.tif	4	100	4
21.tif	5	100	5
22.tif	1	100	1
23.tif	1	100	1
24.tif	12	100	12
25.tif	10	100	10
26.tif	9	100	9
27.tif	6	100	6
28.tif	2	100	2
29.tif	3	100	3
3.tif	3	100	3
30.tif	2	100	2
4.tif	36	97	36
5.tif	19	100	19
6.tif	3	100	3
7.tif	3	100	3
8.tif	3	100	3
9.tif	12	100	12
Average	7.27%	99.90%	7.27%
Standard Dev.	8.21	0.54	8.21