Chapter 4

Multi-Objective GA-Based Image-Adaptive Watermark Embedding for Optimal Fidelity and Robustness

4.1 Introduction

Robust watermarks are designed to resist attempts to remove or destroy the watermark. They are used primarily for copyright protection and content tracking. A number of robust medical image watermarking systems have been developed. For example, one system uses a spread spectrum technique to encode copyright and patient information in images [87]. Another embeds a watermark in a spiral fashion around the Region Of Interest (ROI) of an image [88]. The Gabor transform has also been applied to hide information in medical images [22]. One observation that is generally applicable to robust systems is that attempt to increase robustness of the watermark results in lower image quality [75]. But, watermarked medical images should not differ perceptually from their original counterparts, because the clinical reading of the images (e.g. for diagnosis) must not be affected [22, 75]. However, at the same time, the watermark must be robust against attacks. Thus one of the important requirements of medical image watermarking is to select an optimal embedding strength to make a reasonable trade off-between the two conflicting requirements of fidelity and robustness [89]. The problem becomes more challenging because medical images of different modalities have different noise characters. Even images of same modality under different environments differ in their noise characteristics. Not all watermarking methods are suitable for all image types and all applications [90, 91, 92]. It is of interest to develop a knowledge based approach for automated image-adaptive watermarking [93].

In the last chapter, we investigated the effect of embedding strength on fidelity and robustness by carrying out a number of trial experiments. It was also observed that the maximum level of embedding strength without affecting diagnostic quality differs
for different medical imaging modality. The strength of the watermark was increased or decreased on trial and error computational basis, by running a number of simulations, to achieve PSNR of the watermarked image to be in the range, 38-40 dB. In addition to being a trial and error approach, the algorithm considered only PSNR as the criterion. It would be of interest if the algorithm itself can select the embedding strength based on the input image characteristics. Therefore, in this chapter we propose, in wavelet domain, a GA-based, image adaptive watermarking algorithm, which optimizes the embedding strength to achieve an optimal trade-off between the conflicting goals of fidelity for diagnostics and robustness for security.

Our approach is motivated by the success of GA-based methods in watermarking applications. For example, Pik-Wah Chan method of watermarking optimizes the embedding position of the different parts of a watermark within the video in wavelet domain [94]. Similarly, an attempt has been made to use GA for finding out values of amplitude and modulation index that result in optimal data imperceptibility [67]. Chin-Shiu Shieha et al. have investigated GA-based watermark embedding in DCT domain to find an optimal solution for the conflicting interest of selection of high frequency band for fidelity and low frequency band for robustness [95]. However, this algorithm was tested using only simulated images and no medical images were used for evaluation. Further, for optimal band selection large number of iterations was needed by this algorithm. In this chapter, we propose selection of optimal watermark strength as an optimization problem and use, for the first time, a multi-parameter fitness function for optimal embedding strength selection for public-key, blind medical image watermarking in wavelet domain. The watermarked image is compared with the original one. It is generally acceptable that PSNR values of above 37 dB are visually satisfactory even for the professionals [78]. If the calculated PSNR value lies between 37 and 40 dB, then the process is terminated. Otherwise the embedding strength, $\alpha$ of the watermark is increased or decreased as required and the process is repeated until the desired PSNR is achieved. By this adaptability, each image is watermarked naturally to an extent where optimal level of trade-off between invisibility (PSNR value) and robustness (parameter NC) is achieved.
The algorithm is validated using noise-added Lena images and then evaluated for its potential in teleradiology using three different types of medical images. The algorithm converges within 80 generations and shows different levels of embedding for different modalities. In addition, the use of key concealing and public key encryption (as discussed in the previous chapter) limits the scope of possible attacks, when compared to classic watermarking schemes [90].

4.2 Problem Formulation

Two critical design requirements for medical image watermarking are: i) high image fidelity to meet diagnostic requirements and ii) robustness to conventional attacks as discussed earlier. The watermark should be invisible and robust to intentional or unintentional modification of the image. It is of common opinion that robustness against image distortion is better achieved if the watermark is placed in perceptually significant coefficients of the image [96]. This is argued by the fact that these coefficients do not change much after common image processing and compression operations. Also, if these coefficients are destroyed, the reconstructed image is different from the original image and the digital watermark becomes less meaningful. However, embedding the watermark in perceptually significant coefficients could alter the perceived visual quality of the image. This means that the two basic requirements for an effective watermarking scheme, robustness and invisibility conflict with each other. Traditional watermarking algorithms often solve this problem by choosing parameters via experience (trial and error) which is always inefficient. Conventional search techniques are often incapable of optimizing non-linear functions with multiple variables.

Thus, like most other engineering problems, the design of suitable embedding strength involves multiple, often conflicting, design criteria and specifications. Finding an optimal embedding strength is, therefore, not a simple task. Consequently, there is a need for optimization-based methods that can be used to obtain optimal embedding strength that would satisfy the conflicting requirements. Ideally, the optimization method should lead to the global optimum of the objective function. GA has been widely used to achieve optimal solution in multidimensional nonlinear problem of
conflicting nature [97, 98]. Hence, we resolve the watermarking problem by using GA to obtain the optimized solution based on a multi-objective fitness function. In this chapter image quality indicator, PSNR and robustness indicator, normalized correlation (NC) are used in the design of a novel multi-parameter fitness function to arrive at the optimal embedding strength. A brief introduction to multi-objective GA as applied to solve our problem is presented in the next section.

4.3 Multi-objective Genetic algorithm

A general multi-objective optimization problem consisting of \( k \) competing objectives and \((m+p+q)\) constraints, defined as functions of decision variable set \( x \), can be represented as follows:

Minimize / Maximize \[ f(x) = \left( f_1(x), f_2(x), \ldots, f_m(x) \right) \in Y \]

subject to \( g_i(x) \leq 0, \forall i = 1,2,\ldots,m \)

\[ h_j(x) = 0, \forall j = 1,2,\ldots,p \]

\[ x_l^l \leq x_l \leq x_l^u, \forall l = 1,2,\ldots,q \]

where \( x = (x_1, x_2, \ldots, x_n) \in X \)

Here \( x \) is the decision vector, \( f(x) \) is the multi-objective vector and \( f_j(x) \) is the \( j^{th} \) objective function. \( X \) denotes the decision space and \( Y \) denotes the objective space. The constraints \( g_i(x) \) and \( h_j(x) \) determine the set of feasible solutions. The remaining set of constraints are called variable bounds, restricting each decision variable \( x_l \) to values between a lower \( x_l^l \) and an upper \( x_l^u \) bound. Hard optimization problems typically require many decisions on the input side and many objectives to optimize on the output side. The set of objectives forms a space where points in the space represent individual solutions. The goal is to find the best or optimal solution to the optimization problem at hand.

In an evolutionary algorithm framework, a decision vector naturally corresponds to a candidate solution, and the functions comprising the objective vector are typically incorporated, by various techniques, into the fitness function(s). A dominance test is a way to measure the relative performance among decision vectors. Given two decision vectors \( u \) and \( v \), \( u \) dominates \( v \) if and only if \( a \) ties or exceeds \( v \)’s performance on every
objective, and there exists at least one objective where $u$'s performance strictly exceeds $v$'s. Using this test, one can pare down any given set of decision vectors and find the set of non-dominated decision vectors. A non-dominated set of solutions is formally defined as follows: “a feasible solution to a multi-objective problem is non-dominated if there exists no other feasible solution that will yield an improvement in one objective without causing degradation in at least one other objective” [99]. The set of solutions of a multi-objective optimization problem consists of all decision vectors which cannot be improved in any objective without degradation in the other objectives, the Pareto-optima. Mathematically, the concept of Pareto-optima is as follows. Assuming a minimization problem, the following definitions apply:

**Definition 1 (inferiority)**

A vector $u = (u_1, u_2, ... u_n)$ is said to be inferior to $v = (v_1, v_2, ... v_n)$ if and only if $v$ is partially less than $u$. ($\forall p < u$), i.e., $\forall i = 1,2,...,n, v_i \leq u_i \land \exists i = 1,2,...,n : v_i < u_i$

**Definition 2 (superiority)**

A vector $u = (u_1, u_2, ... u_n)$ is said to be superior to $v = (v_1, v_2, ... v_n)$ if and only if $v$ is inferior to $u$.

**Definition 3 (non-inferiority)**

Vectors $u = (u_1, u_2, ... u_n)$ and $v = (v_1, v_2, ... v_n)$ are said to be non-inferior to one another if $v$ is neither inferior nor superior to $u$.

Pareto optimality defines how to determine the set of optimal solutions. All decision vectors which are not dominated by any other decision vector are called non-dominated or Pareto-optimal. The family of all non-dominated solutions is denoted as Pareto-optimal set (Pareto set) or Pareto-optimal front. An example of a Pareto front is shown in Fig. 4.1.

There are two general approaches to multiple-objective optimization. The first approach is to determine an entire Pareto optimal solution set or a representative subset. But, the size of the Pareto set usually increases with the increase in the number of objectives. There is another method for the multi-criteria optimization, based on the scalarization of the objective function or other non-Pareto approaches. This way the multi-objective problem (MOP) can be easily transformed into a simpler single objective problem (SOP) [52, 100, 101].
Fig. 4.1: Example of Pareto front when minimizing two objective $f_1$ and $f_2$. Nondominated solutions are represented as hollow circles and dominated solutions are shown as filled circles.

4.3.1 Weighted - Sum Approach
This is a simplified alternative approach for our MOP. This method transforms the objective function vector into a higher scalar function using weighted sum of particular objectives. In a single objective optimization, the feasible set is completely ordered according the single objective function. For two solutions $u, v \in X_f$ either $f(u) \geq f(v)$ or $f(v) \geq f(u)$. In case of multi-objective optimization, the feasible set is only partially ordered. In case of minimization this can be formulated as

$$\text{minimize} \ y = f(x) = w_1 f_1(x) + w_2 f_2(x) + \cdots + w_m f_m(x) \quad (4.2)$$

This equation can be modified to the following form (Eqn. 4.3) which can be understood as a sector equation of line with slope $(-w_1/w_2)$ and intercept $(y/w_2)$.

$$f_2(x) = (-w_1/w_2)f_1(x) + y/w_2 \quad (4.3)$$

Thus the problem can be represented in the following form

$$\text{Min} \ \sum_{i=1}^{k} w_i f_i(x) \quad (4.4)$$

subject to $g_i(x) \geq 0, i = 1, 2, ..., m \quad (4.5a)$

$$h_i(x) = 0, i = 1, 2, ..., p \quad (4.5b)$$

where $w_i \geq 0$ are the weighting coefficients. It is usually assumed that $\sum_{i=1}^{k} w_i = 1$. But, these weighting coefficients do not proportionally reflect the relative importance of the objectives, but are only factors which, when varied, locate points in the Pareto set [102]. If one wants $w_i$ to closely reflect the relative importance of the objective
functions, one may normalize the objective functions. This normalization is achieved by using a multiplier $c_i (c_i = 1/f_i)$. After this normalization, the objective function becomes

$$\min \sum_{i=1}^{k} w_i f_i(x) c_i$$  \hspace{1cm} (4.6)

subject to constraints as represented in Eqns. 4.5a and 4.5b.

### 4.4 Proposed System

#### 4.4.1 Methodology

The activity diagram of the GA-based, image-adaptive watermarking algorithm, proposed in this chapter, is presented in Fig. 4.2. The box shown in green color is the module proposed in this chapter for intelligently fine tuning the embedding strength, $\alpha$ to make an optimal trade-off between the two conflicting objectives of fidelity and robustness in medical image watermarking. The rest of the modules are similar to the algorithm presented in the previous chapter. Hence in the following sections the details of the GA-based image adaptive watermark embedding process are presented.

![Activity diagram of the GA-based, image-adaptive, public key-encrypted watermarking scheme](image)

**Fig. 4.2:** Activity diagram of the GA-based, image-adaptive, public key-encrypted watermarking scheme
4.4.2 GA for Optimal Embedding Strength Selection

Encoding Schemes

Problem representation is a critical issue in GA design. Since it affects the performance of a GA significantly, design strategy must select a representational form that is suitable for the specific problem. GAs use various encoding schemes, such as, binary encoding, integer encoding, Gray encoding, and decimal encoding. The choice of encoding is the most important factor in designing a genetic algorithm. The encoding has profound implications on the performance of the GA. Several strategies have been suggested for selecting an encoding scheme. But, until there is more rigorous theory on GAs and the different encodings, the best strategy is to choose an encoding that is naturally suited for the problem at hand, and then design a genetic algorithm that can handle this encoding [50, 103]. For example, binary encoding is useful if the variables of the problem are discrete whereas decimal encoding might be necessary when high-precision is required. As in Holland's original genetic algorithm, binary encoding is the traditional way to represent parameters in most GAs. To use binary encoding with numeric domains the binary representation of a gene \( x_m = [x_{m1}, x_{m2}, \ldots, x_{mn}]^T \) can be mapped (decoded) onto a real number \( a_m \) through a simple linear transformation.

\[
a_m = a_{\text{min}} + \frac{a_{\text{max}} - a_{\text{min}}}{2^{J-1}} \left( \sum_{n=1}^{J-1} x_{mn} 2^{J-1-n} \right) \quad \text{for } m = 1, 2, \ldots, M
\]

(4.7)

where \( a_m \) takes values ranging from \( a_{\text{min}} \) to \( a_{\text{max}} \), and \( x_{mn} \) represents the \( n^{\text{th}} \) bit in the \( m^{\text{th}} \) gene in the binary encoding. In the present study, a 6-bit string is selected to encode the embedding strength. The string representation is then mapped on to a discrete solution space ranging between 0.01 and 0.64. In the present study, the maximum value of the embedding strength was selected to be 0.64 as a further increase in the embedding strength would degrade the watermarked image to a greater extent. The chromosome mapping, used in the present study, is shown in Fig. 4.3.
Conceptually, GAs maintain a population of $N_p$ chromosomes that are selected and created in an iterative process. The population size can be variable, but is usually fixed. The population size at generation $t$ can be denoted as a set of chromosomes as

$$P_t = \{x_{t}(1), x_{t}(2), ..., x_{t}(1N_p)\}$$

(4.8)

To commence the iteration, the GA usually generates a random initial population $P_0$. Other initialization schemes are possible. A priori knowledge of the problem domain is sometimes invoked to seed $P_0$ with good chromosomes. The seed can be obtained by using a classical optimization method. Then, by applying some heuristic technique or through perturbations an initial population, $P_0$ can be generated. The population can also be initialized through a deterministic uniform distribution or by using a combination of two or more of the stated schemes. In the present study the population was initiated by a random process. Once an initial population $P_0$ is created, the main GA cycle can begin.

**Population size**

Determining the size of the population is a crucial factor. Choosing a too smaller population size increases the risk of converging prematurely to a local minimum, since the population may not have enough genetic material to sufficiently cover the problem space. The genetic algorithm may not explore enough of the solution space to
consistently find good solutions. On the other hand, the selection of larger population size to find the global optimum is at the expense of more CPU time. If the rate of genetic change is too high or the selection scheme is chosen poorly, beneficial schema may be disrupted and the population may enter error catastrophe, changing too fast for selection to ever bring about convergence. Hence in this chapter, the suitable population size was arrived by performing a number of trial simulations. These results are discussed in Section 4.5.1.

**Fitness Function**

The fitness function is usually an objective or cost function that can successfully quantify the quality of all possible phenotype solutions. The fitness function is dependent on the environment and application of the system that is undergoing the genetic search process and it is the only connection between the physical problem being optimized and the genetic algorithm itself [104]. Given a population $P_t$ at generation $t$, the GA iteration starts, by evaluating the set

$$F_t = \{f_t(1), x_t(2), \ldots, x_t(1N_p)\}$$

of objective function values associated with the chromosomes $\{\chi_t(k)\}$ with $k = 1,2, \ldots, N_p$. A critical part of any GA optimization problem is the proper selection and implementation of the fitness function.

The main aim of the optimization algorithm is to automatically find out an optimal trade-off between the fidelity of the watermarked image and the robustness of the watermark. As outlined earlier, PSNR of the watermarked image and NC of the recovered watermark can be selected as quantitative metrics for the constraints. In this context, it becomes a multi-objective optimization problem. But, to reduce the complexity, we follow a weighted-sum approach so that the problem can be converted to a single objective problem with a scalar objective function (Eqn. 4.10) [101]. In the present study, the objective function used to evaluate the fitness of each chromosome is selected as given in Eqn. 4.10.

$$f = \max \left( \text{PSNR}(I, I_W) - (\text{wt} \times \text{NC}(W, W')) \right)$$

The conflicting decision vectors, PSNR and NC must be balanced such that no single term dominates the other. If one term is much greater than the other, the GA may
get trapped in a region where that term is acceptable, but the other term is unacceptable, causing the GA to converge slowly or not at all. It is necessary for the fitness function to be well balanced to achieve reasonable results [105]. Because PSNR values are about an order of magnitude larger than the NC values, our results showed that PSNR values were dominating the fitness function. This resulted in a fitness function which was not well balanced. Hence we carried out a series of experiments by increasing the influence caused by NC using various values for weight, wt. Our results suggested a weighting factor of 8 to be optimal to balance the effects caused by the two conflicting requirements.

**Selection Methods**

The selection mechanism in GA essentially defines how the algorithm updates the population from one generation to the next. In general, chromosomes $\chi$ are selected from the population, based on the requirements imposed on the solutions in terms of objective functions $f$ (Eqn. 4.10) to create a new population on the principle of the “survival of the fittest”. This can be achieved by using many different selection methods. But, the most commonly used methods are roulette-wheel, tournament, ranking, and steady-state selection. In the present study, the roulette-wheel selection was used where each individual’s probability of being selected in the next population is proportional to its fitness value. The probability of survival $P_s(k)$ of a chromosome $\chi_t (k)$ is calculated by using the normalized fitness value

$$P_s(i) = \frac{f_r(i)}{\sum_{i=1}^{N_p} f_r(i)}$$

(4.11)

with $\sum_{i=1}^{N_p} P_s(i) = 1$. Since the probability of selection is based on the fitness proportion in the population, this method is also referred to as proportionate selection method.

**GA operators**

The basic GA operators, crossover and mutation, constitute the main algorithm where the population and fitness function can be viewed as external entities.

**Crossover**

Crossover recombines genetic material from selected individuals to form one or more offsprings where some of the useful traits of the parents are preserved. The goal is to
generate new chromosomes that are more fit than their ancestors, thereby contributing to the overall convergence of the population. There are many ways of performing crossover. One-point, two-point, or uniform crossover are used with binary encoding. Arithmetic crossover or perturbation or simulated binary crossover is used with decimal or real encoding.

One-point crossover, illustrated in Fig. 4.4, was used in the present study. During a one-point crossover, two individuals $x(1) = \left[ g_1 g_2 \ldots g_{N_x} \right]^T$ and $x(2) = \left[ g'_1 g'_2 \ldots g'_{N_x} \right]^T$, selected randomly from $P$ undergo crossover if a random number $u$ generated usually from a uniform range of numbers $U \in [0,1]$ is smaller than the probability threshold, $P_x$. Parts of the strings from each individual are swapped at the same location called the crossover point to create two offspring chromosomes $x^c(1)$ and $x^c(2)$ as follows

$$x^c(1) = \left[ g_1 g_2 \ldots g_i g'_{i+1} \ldots g'_{N_x} \right]^T$$

$$x^c(2) = \left[ g'_1 g'_2 \ldots g_i g'_{i+1}, \ldots, g_{N_x} \right]^T$$

The crossover point, $i$ in Eqn 4.12 is chosen randomly from a set of integers

$$I = \{ i \in \mathbb{R} : 1 \leq i \leq N_x - 1 \}$$

In the present study, based on a number of experiments, the crossover probability, optimal $P_x$ was arrived at to be 0.5.

![Fig. 4.4: A typical one-point crossover in binary representation](image)
**Mutation**

Mutation mainly serves to maintain a certain level of genetic diversity in the population to prevent premature convergence of the population to suboptimal values. Bit-flip mutation is the most common mutation operator for binary-encoded GAs. Hence in the present study also, a similar approach was followed. The mutation is realized by simply inverting one or more bits in the chromosome string based on the probability of mutation, \( P_m \). The mutation operator creates a mutated (new) chromosome \( x_m \) from \( x \), as follows.

\[
x^m = [g'_1 g'_2 \ldots g'_{N_x}]^T
\]

where

\[
g'_j = \begin{cases} 
\mu[g_j] & \text{if } P_m < u \in \mathbb{U}(0,1) \\
g_j & \text{otherwise}
\end{cases} \quad \text{with } j = 1, 2, \ldots, N_x
\]

The quantity \( \mu \) in the above equation is a bit-inversion operator that represents the bit flipped from ‘0’ to ‘1’ and vice versa. The binary mutation operation is illustrated in Fig. 4.5. In real-encoded GAs, mutation is generally performed using a perturbation technique similar to that described for crossover, except that the perturbation amount is rather small. A very low rate or no mutation at all will result in premature convergence. As crossover plays the key role in improving the solution, it is assigned a high frequency of occurrence, typically 80-90%. The frequency of occurrence of mutation is kept fairly low, typically 5-10%, to prevent the GA from producing a large number of random solutions.

![Mutation operation in binary representation](image)
Convergence and Halting Conditions

In the analysis of the behavior of genetic algorithms, two types of convergence are to be distinguished. On the one hand, the algorithm is looking for the global optimum, \( f^* \) of the fitness function \( f \). Let \( Z(t) = \max \{ f(x_i) \mid x_i \in x_{1..n}(t) \} \) be the fittest individual present in the population \( x_{1..n}(t) \) at generation \( t \). A GA is said to converge to the global optimum \( f^* \) if and only if

\[
\lim_{t \to \infty} P\{Z(t) = f^*\} = 1 \tag{4.14}
\]

where \( P\{Z(t) = f^*\} \) is the probability that \( Z(t) \) equals \( f^* \).

A second type of convergence is population convergence. Let \( H(x_i, x_j) \) be the Hamming distance between individuals \( x_i \) and \( x_j \), and, with \( n = |x_{1..n}(t)| \) being the population size, and

\[
\overline{H}(t) = \frac{1}{n^2} \sum_{x_i, x_j \in x_{1..n}} H(x_i, x_j) \tag{4.15}
\]

being the average Hamming distance in the population at generation \( t \). The population of the GA is said to converge if and only if

\[
\lim_{t \to \infty} \overline{H}(t) = 0
\]

If the population is converged, all individuals are identical. Based on these concepts, there are two common halting criteria used in GA: i) algorithm can be repeated to a predefined number of generations or ii) it can be repeatedly executed until the convergence of the whole population. In the present work, the second method of halting condition was followed. The last criterion depends critically on the mutation probability. A very high mutation rate will prevent any convergence to occur, and the GA is reduced to random walk. We have investigated the halting condition and these results are presented in section 4.5.3.

4.4.3 GA-based Watermarking Algorithm

Based on these discussions, GA-based algorithm for optimal embedding strength selection was developed. The pseudocode of this algorithm is presented in Table 4.1.
Table 4.1: Pseudocode of the proposed multi-objective GA for optimal embedding strength selection.

\begin{verbatim}
function f = evaluate(P_t)
    \( I_{W}^{LWT} = \text{robust_embed}(I_{W}^{LWT}, W, \infty) \)
    \( I_W = \text{ilwt2}(I_{W}^{LWT}) \)
    \( \text{PSNR}(I_W) = 10 \log_{10} \frac{N \max|I|^2}{\sum_{i=1}^{N}(I-I_w)^2} \)
    \( W' = \text{robust_recover}(I_w, \infty) \)
    \( NC = \frac{<w,w>}{||w'||||w'||} \)
    \( f = \max\left(\text{PSNR}(I,I_W) - (wt * NC(W,W'))\right) \)
    \text{return } f
end

float \( \alpha = \text{function ga_embed(I,W)} \)
    \( I_{W}^{LWT} = \text{ilwt2}(I) \)

\text{Initialize generation, } t=1 \& \text{ generate population } k, P_t = \{P_1, P_2, ..., P_k\}
for \( t=1..k \) in P_t do
    initialize \( P_t \)
    \( f(P_t) = \text{evaluate}(P_t) \)
end for

while halting condition not met do
  Select \( P_1, P_2 \) such that \( P_1, P_2 \in [0,F], \text{ where } F = \text{total fitness} \)
  Select crossover probability \( P_c \)
  Perform crossover
  \( A_{C1} = \text{Crossov}(P_1, P_2, P_c) = (P_1 \cap P_2) \cup (P_1 \cap P_2) \)
  \( A_{C2} = \text{Crossov}(P_1, P_2, P_c) = (P_1 \cap P_2) \cup (P_1 \cap P_2) \)
  Perform mutation with a mutation rate \( P_m \)
  Generate new population such that \( (P_{t+1}, A_c) \geq P_t \)
end while
\end{verbatim}
The activity diagram for the watermarking algorithm is already given in Fig. 4.2. The watermark embedding and detection processes are the same as presented in chapter 3. The pseudocode, given in Table 4.1, was implemented in Matlab.

### 4.5 Selection of GA Parameters

In general, the configurations of GAs are very problem specific. The success of any genetic algorithm largely depends on how well it has been customized for a given application. The customization can be done by choosing proper objective function(s), chromosome encoding scheme, genetic operators, and selection methods. Beside these parameters, there are other parameters and conditions that also affect the performance of a GA. Population size \( P_t \), crossover and mutation probabilities, \( P_c \) and \( P_m \), and termination criteria play significant role in GA's convergence. In spite of many attempts to find the optimal parameter values, systematic trials for specific problems remain the most accepted norm in configuring a GA. Hence experiments were carried out to arrive at optimal GA-parameters for the multi-objective GA-based watermarking of medical images. These results are presented next.

#### 4.5.1 Effect of Chromosome Length

The length of the chromosomes should be optimal, as it increases the likelihood of approaching the global optimum by making the search space more sophisticated. Experiments with 4, 6 and 8-bit long chromosomes were used to investigate the effect of the chromosome length.

The embedding strength, when represented using 4 bits has \( 2^4 = 16 \) possible values. Hence, 16 values are to be selected between 0.1 and 0.64, leading to a quantization error of 0.04. To test the impact of this quantization error, experiments
were performed using the conventional watermarking algorithm by varying $\alpha$ in increments of 0.04. Table 4.2 shows that for a quantization error of 0.04, PSNR varies approximately by 1 dB and NC by 0.01. Increasing the number of bits decreases the quantization error.

<table>
<thead>
<tr>
<th>$\alpha$</th>
<th>Lena</th>
<th>MRI</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSNR</td>
<td>NC</td>
<td>PSNR</td>
</tr>
<tr>
<td>0.04</td>
<td>58.32</td>
<td>0.84</td>
</tr>
<tr>
<td>0.08</td>
<td>57.86</td>
<td>0.85</td>
</tr>
<tr>
<td>0.12</td>
<td>55.09</td>
<td>0.85</td>
</tr>
<tr>
<td>0.16</td>
<td>53.14</td>
<td>0.87</td>
</tr>
<tr>
<td>0.20</td>
<td>52.62</td>
<td>0.89</td>
</tr>
<tr>
<td>0.24</td>
<td>51.53</td>
<td>0.90</td>
</tr>
<tr>
<td>0.28</td>
<td>50.49</td>
<td>0.91</td>
</tr>
<tr>
<td>0.32</td>
<td>49.76</td>
<td>0.92</td>
</tr>
</tbody>
</table>

For 6-bit encoding, the quantization error reduces to 0.01. Table 4.3 shows that there is no significant difference between PSNR and NC values when embedding strength is varied in increments of 0.01. An 8-bit chromosome representation will further reduce the quantization error, however the computational time goes high as shown in Fig. 4.6. Hence, as a trade-off, 6-bit representation was selected for the present study.

<table>
<thead>
<tr>
<th>$\alpha$</th>
<th>Lena</th>
<th>MRI</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSNR</td>
<td>NC</td>
<td>PSNR</td>
</tr>
<tr>
<td>0.01</td>
<td>58.64</td>
<td>0.83</td>
</tr>
<tr>
<td>0.02</td>
<td>58.57</td>
<td>0.83</td>
</tr>
<tr>
<td>0.03</td>
<td>58.41</td>
<td>0.83</td>
</tr>
<tr>
<td>0.04</td>
<td>58.32</td>
<td>0.84</td>
</tr>
<tr>
<td>0.05</td>
<td>58.26</td>
<td>0.84</td>
</tr>
<tr>
<td>0.06</td>
<td>58.09</td>
<td>0.84</td>
</tr>
<tr>
<td>0.07</td>
<td>57.95</td>
<td>0.84</td>
</tr>
<tr>
<td>0.08</td>
<td>57.86</td>
<td>0.85</td>
</tr>
</tbody>
</table>
4.5.2 Effect of Population Size on Convergence

As pointed out earlier, population-size also contributes to optimal performance of GA. A smaller size population reduces the evolution cost, but results in premature convergence, because the population provides insufficient samples in the search space. However, more computing time is needed for a large-size population. Hence, it is desirable to arrive at a suitable population size. A few additional experiments were performed to find appropriate population size using fixed crossover rate, mutation rate and stopping condition. Fig. 4.7 shows the results (trial experiments) to investigate the effect of population size on the convergence of fitness function. For population size, 20 the convergence is faster. For population size, 60 it is slow, but there is no significant difference in the fitness function value as compared to that of population size, 40.

One of the limitations of GA is its computational overhead. Hence, the execution time of the GA-based embedding strength selection was computed by varying the population size. These results, presented in Fig. 4.8 show a fast increase in execution time as the population size increases above 50.
Based on the results presented in Fig. 4.7 and Fig. 4.8, one can conclude that for a gain of 0.04 in fitness value, relatively large amount of execution time, (100 sec) is needed. Hence a population size of 40 was selected as optimal parameter for further experiments.
4.5.3 Halting Condition

Two types of termination criteria were followed. The first criterion was based on the number of generations (referred to as "convergence on generation"). To arrive at this criterion the average distance between the chromosomes were computed as a function of number of generations. These results are presented in Fig. 4.9. After an oscillatory behavior, the system converges at 120\textsuperscript{th} generation. Therefore, the algorithm was terminated upon reaching 120\textsuperscript{th} generation. The second criterion was based on the convergence of the best fitness score in the population (referred to as "convergence on best score") wherein the average distance between chromosomes was less than 0.01. In the present study, the GA was terminated when either one of these two conditions was met. The various GA parameters used in this study were thus optimized and these values are given in Table 4.4.

![Fig. 4.9: Influence of number of generation on halting condition (population size 40)](image)

4.6 Performance Analysis

For validation of the algorithm, the same six images (three Lena and three medical) and the two different size watermarks, described in chapter 3.7 were used. The performance
of the GA-based watermarking was evaluated through several experiments with these two different levels of payload.

<table>
<thead>
<tr>
<th>Genetic Operator</th>
<th>Optimization parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial population</td>
<td>40 chromosomes</td>
</tr>
<tr>
<td>Encoding</td>
<td>6-bit binary encoding scheme</td>
</tr>
<tr>
<td>Fitness function</td>
<td>Multi-objective as specified in Table 4.10</td>
</tr>
<tr>
<td>Selection</td>
<td>Roulette wheel selection</td>
</tr>
<tr>
<td>Crossover</td>
<td>Single-point crossover with random crossover probability 0.5</td>
</tr>
<tr>
<td>Mutation</td>
<td>Single-point mutation with probability of 0.062</td>
</tr>
<tr>
<td>Convergence</td>
<td>Convergence to single result for the whole population or 120 generations</td>
</tr>
</tbody>
</table>

### 4.6.1 Validation Using Lena Images

Fig. 4.10 presents the results of the GA-based algorithm evaluated using the Lena images. There is no distortion in the perceptual quality of the watermarked images. The embedding strengths used for watermarking of these six images were automatically arrived at by the algorithm. The various GA-parameters used in these experiments are presented in Table 4.4.

Watermarking experiments were also performed using $\alpha$ values selected manually. The PSNR and NC values computed from the watermarked Lena images, obtained from the trial and error basis non-GA method and the multi-objective GA approach are collected in Table 4.5. These results clearly show the significance of the GA-based approach. For example, the embedding strength selected for the Lena image is 0.2 in case of the non-GA approach. However, an image with good PSNR, like the Lena image, can withstand a high embedding strength. The proposed GA-based watermarking scheme adaptively varies the embedding strength depending upon the PSNR of the input image, without compromising on the image fidelity and robustness of the watermark, as seen from an $\alpha$ value of 0.61 for Lena image in Table 4.5. Similar experiments were also done for high payload embedding and the quantitative
parameters are given in Table 4.5. For the high payload, the $\alpha$ value automatically goes down from 0.61 to 0.32, whereas for the non-GA, an $\alpha$ of 0.1 was arbitrarily used. For noisy Lena-M, the multi-objective GA automatically fine tunes the $\alpha$ to 0.17 for high payload to retain PSNR above the required 37 dB.

![Visual evaluation of the GA-based algorithm. The input images are the same as those in column 1 of Fig. 3.6](image-url)
4.6.2 Application to Medical Images

With the success of the multi-objective GA to automatically fine tune $\alpha$ in noise-modeled Lena images, medical image watermarking was performed next. These results are presented in Fig. 4.11 for visual comparison and the quantitative parameters are collected in Table 4.5. One can very readily notice that, $\alpha$ values are different for the three different medical images and these values need not be specified by the user apriori. Thus multi-objective GA address the challenge of optimal trade off between the conflicting PSNR and NC, and fine tunes $\alpha$ to keep PSNR > 37 dB and NC above 0.85.

Table 4.5: Comparison of PSNR and NC values based on trial and error approach and GA-based approach for low and high payloads.

<table>
<thead>
<tr>
<th>Payload</th>
<th>Image</th>
<th>Non-GA Embedding</th>
<th>GA-based Embedding</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$\alpha$</td>
<td>PSNR</td>
</tr>
<tr>
<td>Low Payload</td>
<td>Lena</td>
<td>0.2</td>
<td>52.62</td>
</tr>
<tr>
<td></td>
<td>Lena - A</td>
<td>0.2</td>
<td>40.15</td>
</tr>
<tr>
<td></td>
<td>Lena - M</td>
<td>0.2</td>
<td>39.75</td>
</tr>
<tr>
<td></td>
<td>Ultrasound</td>
<td>0.2</td>
<td>38.73</td>
</tr>
<tr>
<td></td>
<td>Fundus</td>
<td>0.2</td>
<td>41.64</td>
</tr>
<tr>
<td></td>
<td>MRI</td>
<td>0.2</td>
<td>39.23</td>
</tr>
<tr>
<td>High Payload</td>
<td>Lena</td>
<td>0.1</td>
<td>49.23</td>
</tr>
<tr>
<td></td>
<td>Lena - A</td>
<td>0.1</td>
<td>40.19</td>
</tr>
<tr>
<td></td>
<td>Lena - M</td>
<td>0.1</td>
<td>39.81</td>
</tr>
<tr>
<td></td>
<td>Ultrasound</td>
<td>0.1</td>
<td>41.52</td>
</tr>
<tr>
<td></td>
<td>Fundus</td>
<td>0.1</td>
<td>42.63</td>
</tr>
<tr>
<td></td>
<td>MRI</td>
<td>0.1</td>
<td>39.48</td>
</tr>
</tbody>
</table>

4.6.3 Robustness to Attacks

In real world applications, when the watermarked medical images are distributed on a public network, it is encountered by different types of attacks. Hence the GA-based watermarked images were subjected to different types of attacks, resulting in tampered
images. The retrieval logic was implemented on the attacked images to recover the watermarks. Table 4.6 shows the survival of the watermark from watermarked MRI image that was subjected to different type of attacks. It is seen that the proposed scheme withstands many common attacks. However, for high compression ratio (70%), the recovery of the watermark is not very useful.

Fig. 4.11: Visual evaluation using medical images. The details of the input images are the same as in column 1 of Fig. 3.7
<table>
<thead>
<tr>
<th>Attack</th>
<th>Low Payload</th>
<th></th>
<th>High Payload</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Recovered Watermark</td>
<td>NC</td>
<td>Recovered Watermark</td>
<td>NC</td>
</tr>
<tr>
<td>Transmission</td>
<td>0.96</td>
<td></td>
<td>0.97</td>
<td></td>
</tr>
<tr>
<td>Copy</td>
<td>0.96</td>
<td></td>
<td>0.97</td>
<td></td>
</tr>
<tr>
<td>Rotation (40)</td>
<td>0.93</td>
<td></td>
<td>0.94</td>
<td></td>
</tr>
<tr>
<td>Filtering (Averaging)</td>
<td>0.88</td>
<td></td>
<td>0.92</td>
<td></td>
</tr>
<tr>
<td>Compression (40%)</td>
<td>0.86</td>
<td></td>
<td>0.87</td>
<td></td>
</tr>
<tr>
<td>Compression (70%)</td>
<td>0.76</td>
<td></td>
<td>0.62</td>
<td></td>
</tr>
</tbody>
</table>
4.6.4 Computational Time

A major concern of the GA-based approach is the time complexity. At present the total execution time for watermarking a 507 x 481 MRI image, using an Intel dual core 2GHz processor with 1GB RAM, under windows XP platform was estimated to be about 6.23 minutes for the low payload. Watermarking the same image with a high payload took 9.58 minutes. The computational time is calculated without any code optimization. The embedding and extraction of watermark took 6.17 sec and 3.23 sec, respectively. Hence the embedding strength selection by GA takes the major execution time. However, this problem can be circumvented to some extent exploiting the parallel nature of GA.

4.7 Conclusion

An innovative multi-objective algorithm for optimal embedding strength selection to achieve conflicting goals of fidelity and robustness in blind medical image watermarking in wavelet domain is developed. In this GA-based watermarking algorithm, the fidelity parameter, PSNR and the robustness parameter, NC are used to design the multi-objective fitness function. Experiments using different types of medical images show that the GA-optimized embedding strength depends on the input image characteristics. In addition, it is also observed that the magnitude of embedding strength is different for different imaging modalities. An interesting result of this study is the insight into the trade-off between fidelity and robustness. A multi-objective problem is reduced to the simple and computationally efficient single object problem using the weighted sum approach. Study of all possible Pareto-optimal solutions is planned for a future investigation.