IMAGE ENHANCEMENT USING EVOLUTIONARY COMPUTING

MAJOR PROJECT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF DEGREE OF

Master of Technology

In

Information Systems

Submitted By:

KRITI GUPTA

(2K12/ISY/15)

Under the Guidance of

Dr. O. P. Verma

(HOD, Department of IT)

DEPARTMENT OF INFORMATION TECHNOLOGY

DELHI TECHNOLOGICAL UNIVERSITY

(2012-2014)

ABSTRACT

A new method for the enhancement of color images is presented which uses the fuzzy logic and differential evolution. With the application of an objective measure known as exposure the image is divided into underexposed and overexposed regions. The V component or the luminance component of the HSV color space is exploited for the enhancement process. However, the hue component is kept intact so that the color constitution of the original image remains the same. The under exposed and the over exposed regions are enhanced using two different schemes. The enhancement of the underexposed region is done using a parametric sigmoid function whereas the enhancement of the overexposed region is done using power law transformation. For good enhancement results, the appropriate values of the parameters of the sigmoid function and the gamma used in power law transformation function are required. These are optimized using differential evolution. Moreover, two separate membership functions are used to characterize the underexposed and the over exposed regions of the image, i.e., Gaussian membership for the underexposed areas and triangular membership for the overexposed areas. This method becomes universal for implementation upon all types of contrast degradations because of the use of separate membership functions and enhancement operators for the two regions. The self-adaptation of the parameters, based on differential evolution, makes the whole method automatic.

ACKNOWLEDGEMENT

I take the opportunity to express my sincere gratitude to my project mentor Dr. O.P. Verma, Head of Department, Department of Information Technology, Delhi Technological University, Delhi for providing valuable guidance and constant encouragement throughout the project. It is my pleasure to record my sincere thanks to him for his constructive criticism and insight without which the project would not have shaped as it has.

I humbly extend my words of gratitude to other faculty members of this department for providing their valuable help and time whenever it was required.

Kriti Gupta
Roll No. 2K12/ISY/15
M.Tech (Information Systems)
E-mail: kriti311290@gmail.com

CERTIFICATE

This is to certify that the thesis entitled, **"Image Enhancement using Evolutionary Computing"** submitted by **Kriti Gupta (2K12/ISY/15)** in partial fulfillment of the requirements for the award of Master of Technology Degree in Information Systems during session 2012-2014 at Delhi Technological University is an authentic work carried out by her under my supervision and guidance.

To the best of my knowledge, the matter embodied in the thesis has not been submitted to any other University / Institute for the award of any Degree or Diploma.

Dr. O. P. Verma
Head of Department
Department of Information Technology
Delhi Technological University
Delhi

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
INTRODUCTION

1.1 OVERVIEW

Image enhancement is needed to compensate for the limitations of the hardware capturing the image. Also, many a times the quality of an image suffers due to lack of proper lighting and illumination. Thus, to improve the visual perception of an image and to unveil the information in an image, which is not otherwise obvious, image enhancement is performed. Image enhancement can be broadly classified into two sections according to the domain of working i.e., spatial domain and frequency domain. Enhancement using spatial domain manipulates directly the pixel intensities. For working in frequency domain the image is first converted into frequency domain using Fourier transform and then operated upon for enhancement, and then converted back to original form. Due to the ease of implementation, spatial domain enhancement has attracted greater attention than frequency domain techniques.

Since the process of image enhancement changes the appearance of an image, some precautions should be taken during the process. The original color composition (hue) should not be affected. The value of other components should not exceed the maximum value in the image [1]. Also, the noise present in the transformed image should not be amplified with respect to the original image, if not damped. In all, the visual perception of the output image should be better than the input image but the other visual factors should not be compromised.

Exposure based enhancement techniques perform better than simple enhancement techniques. Due to striking contrast between the nature of under exposed region and over exposed region, if enhancement is done according to the intensity exposition i.e., if the under exposed and the over exposed regions are enhanced using two different schemes, the resulting image is better. Here, the enhancement of the underexposed region is done using a parametric sigmoid function whereas the enhancement of the overexposed region is done using power law transformation.

Appropriate parameter selection is important for success of any algorithm. Thus, the values of the parameters of the sigmoid function and the gamma used in power law transformation function are optimized using differential evolution. Differential Evolution (DE) algorithm is a heuristic approach mainly having three advantages; finding a true global minimum of a multi modal search space regardless of the initial parameter values; fast convergence rate and using a few control parameters [2].

1.2 PROBLEM STATEMENT

The motivation is to develop an automatic as well as efficient algorithm for enhancement of color images of mixed type (i.e., containing both under exposed and over exposed regions). Here, automatic means that it requires no human intervention for any parametric settings i.e., once an image is provided, the algorithm takes all decision on its own to give the best possible results. And efficient means that the algorithm gives better output results than the existing works, i.e., better visual perception, lesser computational time and lesser algorithmic complexity.

1.3 PROPOSED SOLUTION

The problem of color image enhancement can be solved by using a technique based on intensity exposition. The process of enhancement starts with assigning each pixel a membership value to the under exposed and the over exposed region. A Gaussian membership function and a triangular membership function is used for underexposed region and over exposed region respectively. The Shannon entropy of the image is calculated based on these memberships. The under exposed and the over exposed regions are enhanced using parametric sigmoid function and power law operator respectively. An objective function is formulated based on the Shannon entropy of the image, which is minimized by the differential evolution algorithm such that the parameters of both the enhancement operators are optimized. Once the appropriate parameter selection is done, defuzzification is performed to get the final enhanced image. The results of this algorithm are compared with an algorithm that uses bacterial foraging algorithm and the concept of visual factors [1].

CHAPTER 2
LITERATURE REVIEW

2.1 OPTIMISATION PROBLEM AND ALGORITHM

An optimization problem is the problem of finding the *best* solution from all feasible solutions. The *standard form* of a (continuous) optimization problem is as given by equation (1) and is shown below:

$$\begin{aligned} &\underbrace{\text{minimize}}_{x}\ f(x) \\ &\text{subject to}\ \ g_i(x) \le 0, \quad i = 1, ..., m \\ &\qquad\qquad h_i(x) = 0, \quad i = 1, ..., p \end{aligned} \tag{1}$$

Where

$f(x): \mathrm{R}^n \to R$ is the objective function to be minimized over the variable x,

$g_i(x) \le 0$ are called inequality constraints, and

$h_i(x) = 0$ are called equality constraints.

By convention, the standard form defines a minimization problem. A maximization problem can be treated by negating the objective function.

An optimization problem has basically three components: a function to optimize, possible solution set to choose a value for the variable from, and the optimization rule, which will be either maximized or minimized.

An optimization problem is a problem of finding values for the variables of a function to optimize the function. These kinds of problems exist in many disciplines. Whenever a decision needs to be made and the problem is formulated using mathematical terms, optimization algorithms will be used to solve the formulated problem.

There are 5 broad types of optimizations:

i. Combinatorial Optimization
ii. Dynamic Programming
iii. Evolutionary Algorithms
iv. Gradient Method
v. Stochastic Optimization

The algorithm used for a problem is chosen depending on the behavior of the problem. For example, if both the objective function and the functions which construct the feasible region are linear, it is called a linear programming problem (subcategory of combinatorial optimization), and methods like simplex algorithm will be used to solve it. Evolutionary algorithm will be discussed in detail in the next subsection.

2.2 EVOLUTIONARY ALGORITHM

An evolutionary algorithm (EA) is a meta-heuristic optimization algorithm using techniques inspired by mechanisms from organic evolution such as mutation, recombination, and natural selection to find an optimal configuration for a specific system within specific constraints. These are closely linked to AI techniques, especially search techniques and can be regarded as population-based stochastic generate-and-test algorithms.

Most of these algorithms are inspired by biological aspects. Unlike deterministic solution methods, meta-heuristic algorithms are not affected by the behavior of the optimization problem. This makes the algorithm to be used widely in different fields. Since the introduction of genetic algorithm in 1975, many meta-heuristic algorithms are introduced.

Evolutionary algorithms are inspired by Darwinian's theory of evolution or the principle of natural selection. Natural selection is the gradual process by which biological traits become either more or less common in a population as a function of the effect of inherited traits on the differential reproductive success of organisms interacting with their environment.

2.2.1 Components of Evolutionary Algorithms

a) Representation

Objects forming possible solutions within the original problem context are referred as phenotypes, their encoding; the individuals within the Evolutionary Algorithm are called genotypes. The first design step is commonly called Representation, as it amounts to specifying a mapping from the phenotypes onto a set of genotypes that are said to represent these phenotypes.

b) Evaluation Function

The role of evaluation function is to represent the requirements to adapt to. It forms the basis of selection, and thereby it facilitates improvements. It is a function or procedure that assigns a quality measure to genotypes. The evaluation function is commonly called the Fitness function.

c) Population

The role of population is to hold possible solutions. A population is a multiset of genotypes. Defining a population can be as simple as specifying how many individuals are in it, that is, setting the population size.

d) Parent Search Mechanism

The role of parent selection or matching selection is to distinguish among individuals based on their quality, in particular, to allow the better individuals to become parents of the next generation. Together with survivor selection mechanism, parent selection is responsible for pushing quality improvements.

e) Variation Operators

Its role is to create new individuals from old ones. It performs the generate step.

- **Mutation**

  A unary variation operator is commonly called mutation. A mutation operator is always stochastic. Its output depends on the outcomes of a series of random choices.

- **Recombination**

A binary variation operator is called as Recombination or crossover. Similar to mutation it is stochastic operator.

f) Survivor Selection Mechanism

The role of survivor selection mechanism or environmental selection is to distinguish among individuals based on their quality. As opposed to parent selection which is typically stochastic, survivor selection is often deterministic.

- **Initialization**

The first population is seeded by randomly generated individuals. In principle, problem specific heuristics can be used in this step aiming at an initial population with higher fitness.

- **Termination Condition**

If the problem has a known optimal fitness level, probably coming from a known optimum of the given objective function, then reaching this level should be used as stopping condition. However Evolutionary Algorithms are stochastic and mostly there are no guarantees to reach an optimum, hence this condition might never get satisfied and the algorithm may never stop. This requires that this condition is extended with one that certainly stops the algorithm. Commonly used options for this purpose are:

a) The total number of fitness evaluations reaches a given limit.
b) The population diversity drops under a given threshold.
c) The maximally allowed CPU time elapses.
d) For a given period of time, the fitness improvements remain under a threshold value.

2.2.2 Evolutionary Algorithm Types

a) Genetic Algorithm

This is the most popular type of EA. One seeks the solution of a problem in the form of strings of numbers (traditionally binary, although the best representations are usually those that reflect

something about the problem being solved), by applying operators such as recombination and mutation (sometimes one, sometimes both). This type of EA is often used in optimization problems.

b) Genetic Programming

Here the solutions are in the form of computer programs, and their fitness is determined by their ability to solve a computational problem.

c) Evolutionary Programming

Similar to genetic programming, but the structure of the program is fixed and its numerical parameters are allowed to evolve.

d) Gene Expression Programming

Like genetic programming, GEP also evolves computer programs but it explores a genotype-phenotype system, where computer programs of different sizes are encoded in linear chromosomes of fixed length.

e) Evolution Strategy

Works with vectors of real numbers as representations of solutions, and typically uses self-adaptive mutation rates.

f) Differential Evolution

Based on vector differences and is therefore primarily suited for numerical optimization problems.

g) Neuroevolution

Similar to genetic programming but the genomes represent artificial neural networks by describing the structure and connection weights. The genome encoding can be direct or indirect.

h) Learning Classifier System

Here the solutions are classifiers (rules or conditions). A Michigan-LCS works with individual classifiers whereas a Pittsburgh-LCS uses populations of classifier-sets. Initially, classifiers were

only binary, but now include real, neural network or S-expression types. Fitness is determined with either strength or accuracy based reinforcement learning approach.

i) Swarm Algorithms

Swarm algorithm is inspired by the foraging or breeding or flocking behavior of swarms.

2.2.3 Basic Evolutionary Algorithm

- Generate the initial population of individuals randomly - first Generation
- Evaluate the fitness of each individual in that population
- Repeat on this generation until termination (time limit, sufficient fitness achieved, etc.)
    - Select the best-fit individuals for reproduction - parents
    - Breed new individuals through crossover and mutation operations to give birth to offspring
    - Evaluate the individual fitness of new individuals
    - Replace least-fit population with new individuals

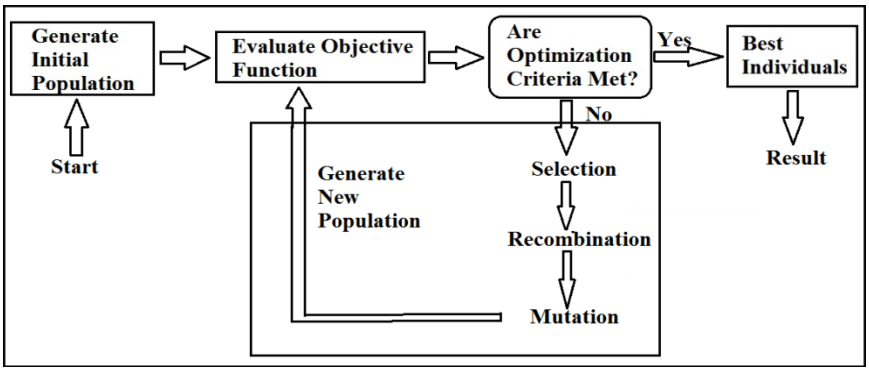

Figure 2-1 Basic evolutionary algorithm

2.2.4 Advantages of Evolutionary Algorithm

a) Conceptual Simplicity

A primary advantage of evolutionary computation is that it is conceptually simple. The algorithm consists of initialization, which may be a purely random sampling of possible solutions, followed by iterative variation and selection in light of a performance index. This figure of merit must

assign a numeric value to any possible solution such that two competing solutions can be rank ordered. Finer granularity is not required. Thus the criterion need not be specified with the precision that is required of some other methods. In particular, no gradient information needs to be presented to the algorithm. Over iterations of random variation and selection, the population can be made to converge to optimal solutions.

b) Broad Applicability

Evolutionary algorithms can be applied to virtually any problem that can be formulated as a function optimization task. It requires a data structure to represent solutions, a performance index to evaluate solutions, and variation operators to generate new solutions from old solutions (selection is also required but is less dependent on human preferences). The state space of possible solutions can be disjoint and can encompass infeasible regions, and the performance index can be time varying, or even a function of competing solutions extant in the population.

c) Outperform Classic Methods on Real Problems

Real-world function optimization problems often (1) impose nonlinear constraints, require payoff functions that are not concerned with least squared error, (3) involve non-stationary conditions, (4) incorporate noisy observations or random processing, or include other vagaries that does not conform well to the prerequisites of classic optimization techniques. The response surfaces posed in real-world problems are often multimodal, and gradient-based methods rapidly converge to local optima (or perhaps saddle points) which may yield insufficient performance.

2.2.5 Disadvantages of Evolutionary Algorithm

a) There is no guarantee for finding optimal solutions in a finite amount of time.

- However, asymptotic convergence proofs are available.
- For specific problems, computational complexity worked out.

b) Success of almost every evolutionary algorithm depends on appropriate parameter selection however parameter tuning is done mostly by trial-and-error.

- Self-adaptation is a remedy.

c) Population approach may be expensive, computationally.

- Parallel implementation is a remedy

2.2.6 Bacterial Foraging Optimization (BFO)

Bacterial foraging optimization algorithm (BFOA), proposed by Passino, has been broadly acknowledged as a global optimization algorithm. The key idea of this algorithm is inspired by the social foraging behavior of *Escherichia coli* bacteria in multi-optimal function optimization.

During foraging of the real bacteria, locomotion is achieved by a set of tensile flagella. Flagella help an E.coli bacterium to tumble or swim, which are two basic operations performed by a bacterium at the time of foraging. Moving the flagella in the counterclockwise direction helps the bacterium to swim at a very fast rate. The bacteria undergoes chemotaxis, where they like to move towards a nutrient gradient and avoid noxious environment. Generally the bacteria move for a longer distance in a friendly environment.

When they get food in sufficient, they are increased in length and in presence of suitable temperature they break in the middle to from an exact replica of itself. This phenomenon inspired Passino to introduce an event of reproduction in BFOA. Due to the occurrence of sudden environmental changes or attack, the chemotactic progress may be destroyed and a group of bacteria may move to some other places or some other may be introduced in the swarm of concern. This constitutes the event of elimination-dispersal in the real bacterial population, where all the bacteria in a region are killed or a group is dispersed into a new part of the environment.

Now suppose that we want to find the minimum of $J(\theta)$ where $\theta \in \Re^p$ (i.e. θ is a p-dimensional vector of real numbers), and we do not have measurements or an analytical description of the gradient $\nabla J(\theta)$. BFOA mimics the four principal mechanisms observed in a real bacterial system: chemotaxis, swarming, reproduction, and elimination-dispersal to solve this non-gradient optimization problem. A virtual bacterium is actually one trial solution (may be called a search-agent) that moves on the functional surface to locate the global optimum.

Let j be the index for the chemotactic step. Let k be the index for the reproduction step. Let l be the index of the elimination-dispersal event. Also let the following terms as:

p : the dimension of the search space

S : the number of bacteria in the population iterated by counter i

N_c : the number of chemotactic steps iterated by counter j

N_s : the number of swims after tumble iterated by the counter m

N_{re} : the number of reproductive steps iterated by counter k

N_{ed} : the number of elimination dispersal events iterated by the counter l

p_{ed} : elimination dispersal probability

$C(i,k)$: the size of step taken in a random direction specified by tumble

Let $p(j,k,l)=\left\{\theta^i(j,k,l)\middle| i=1,2,...,S\right\}$ represent the position of each member in the population of the S bacteria at the j^{th} chemotactic step, k^{th} reproduction step, and l^{th} elimination-dispersal event. Here, let $J(i,j,k,l)$ denote the cost at the location of the i^{th} bacterium $\theta^i(j,k,l)\in\Re^p$

a) Chemotaxis

Bacterium can swim for a period of time in the same direction or it may tumble, and alternate between these two modes of operation for the entire lifetime. The movement of the bacterium may be represented by

$$\theta^i(j+1,k,l)=\theta^i(j,k,l)+C(i)\frac{\Delta(i)}{\sqrt{\Delta^T(i)\Delta(i)}} \tag{2}$$

Where Δ indicates a vector in the random direction.

b) Swarming

The cell-to-cell signaling in E. coli swarm may be represented by the following function.

$$J_{cc}\left(\theta, P\left(j,k,l\right)\right) = \sum_{i=1}^{S} J_{cc}\left(\theta, \theta^{i}\left(j,k,l\right)\right)$$

$$= \sum_{i=1}^{S}\left[-d_{\text{attractant}} \exp\left(-w_{\text{attractant}} \sum_{m=1}^{p}\left(\theta_m - \theta^{i}{}_m\right)^2\right)\right] + \sum_{i=1}^{S}\left[h_{\text{repellant}} \exp\left(-w_{\text{repellant}} \sum_{m=1}^{p}\left(\theta_m - \theta^{i}{}_m\right)^2\right)\right] \quad (3)$$

Where $J_{cc}\left(\theta, P\left(j,k,l\right)\right)$ is the objective function value to be added to the actual objective function (to be minimized) to present a time varying objective function.

$d_{\text{attractant}}$, $w_{\text{attractant}}$, $h_{\text{repellant}}$, $w_{\text{repellant}}$ are different coefficients.

c) Reproduction

The least healthy bacteria eventually die while each of the healthier bacteria (those yielding lower value of the objective function) asexually split into two bacteria, which are then placed in the same location. This keeps the swarm size constant.

d) Elimination and Dispersal

A significant local rise of temperature may kill a group of bacteria that are currently in a region with a high concentration of nutrient gradients. Events can take place in such a fashion that all the bacteria in a region are killed or a group is dispersed into a new location. To simulate this phenomenon in BFOA some bacteria are liquidated at random with a very small probability while the new replacements are randomly initialized over the search space.

2.3 DIFFERENTIAL EVOLUTION

In 1995, a new floating point encoded an evolutionary algorithm for global optimization, called Differential Evolution (DE) was proposed by R. Storn and K. V. Price. Eventually, DE has turned out to be the best evolutionary algorithm for solving the real-valued test function suite.

When the cost function is nonlinear and non-differentiable, direct search approaches are the methods of choice. Central to every direct search method is a strategy that generates variations of the parameter vectors. Once a variation is generated, a decision must then be made whether or not to accept the newly derived parameters. Most standard direct search methods use the greedy criterion to make this decision. Under the greedy criterion, a new parameter vector is accepted if

and only if it reduces the value of the cost function. Although the greedy decision process converges fairly fast, it runs the risk of becoming trapped in a local minimum.

Inherently parallel search techniques like genetic algorithms and evolution strategies have some built-in safeguards to forestall misconvergence. By running several vectors simultaneously, superior parameter configurations can help other vectors escape local minima.

Another method which can extricate a parameter vector from a local minimum is Simulated Annealing: Annealing relaxes the greedy criterion by occasionally permitting an uphill move. Such moves potentially allow a parameter vector to climb out of a local minimum. As the number of iterations increases, the probability of accepting a large uphill move decreases. In the long run, this leads to the greedy criterion.

Users generally demand that a practical minimization technique should fulfil following requirements:

(a) Ability to handle non-differentiable, nonlinear and multimodal cost functions.

(b) Parallelizability to cope with computation intensive cost functions.

(c) Ease of use, i.e. few control variables to steer the minimization. These variables should also be robust and easy to choose.

(d) Good convergence properties, i.e. consistent convergence to the global minimum in consecutive independent trials.

Differential Evolution (DE) was designed to fulfil all of the above requirements [3]. To fulfil requirement (a) DE was designed to be a stochastic direct search method. Direct search methods also have the advantage of being easily applied to experimental minimization where the cost value is derived from a physical experiment rather than a computer simulation. Requirement (b) is important for computationally demanding optimizations where, for example, one evaluation of the cost function might take from minutes to hours, as is often the case in integrated circuit design or finite element simulation. In order to obtain usable results in a reasonable amount of time, the only viable approach is to resort to a parallel computer or a network of computers. DE fulfils requirement (b) by using a vector population where the stochastic perturbation of the population vectors can be done independently. In order to satisfy requirement (c) it is advantageous if the minimization method is self-organizing so that very little input is required

from the user. The good convergence properties demanded in requirement (d) are mandatory for a good minimization algorithm. Although many approaches exist to theoretically describe the convergence properties of a global minimization method, only extensive testing under various conditions can show whether a minimization method can fulfil its promises [3].

In DE community, the individual trial solutions, which constitute a population, are called *parameter vectors* or *genomes*. DE operates through the same computational steps as employed by a standard EA. However, unlike traditional EAs, DE employs difference of the parameter vectors to explore the objective function landscape.

The performance of DE primarily depends on the mutation strategy, the crossover operation, and the intrinsic control parameters like scale factor (F), crossover rate (Cr), and population size (N_P). The DE family now consists of more than five distinct mutation strategies, some of which like DE/current-to-best/1, may even generate mutated recombinants and two prominent crossover schemes i.e. exponential and binomial. Each of these mutation and crossover operations may be effective over certain problems but poorly perform over the others.

DE searches for a global optimum point in a D-dimensional real parameter space R^D . It begins with a randomly initiated population of NP D dimensional real-valued parameter vectors. Each vector, also known as genome/chromosome, forms a candidate solution to the multidimensional optimization problem. NP does not change during the minimization process. The initial vector population is chosen randomly and should cover the entire parameter space. As a rule, we will assume a uniform probability distribution for all random decisions unless otherwise stated. In case a preliminary solution is available, the initial population might be generated by adding normally distributed random deviations to the nominal solution.

DE generates new parameter vectors by adding the weighted difference between two population vectors to a third vector. This operation is called mutation.

The mutated vector's parameters are then mixed with the parameters of another predetermined vector, the target vector, to yield the so-called trial vector. Parameter mixing is often referred to as crossover in the ES-community and will be explained later in more detail.

If the trial vector yields a lower cost function value than the target vector, the trial vector replaces the target vector in the following generation. This last operation is called selection. Each population vector has to serve once as the target vector so that NP competitions take place in one generation.

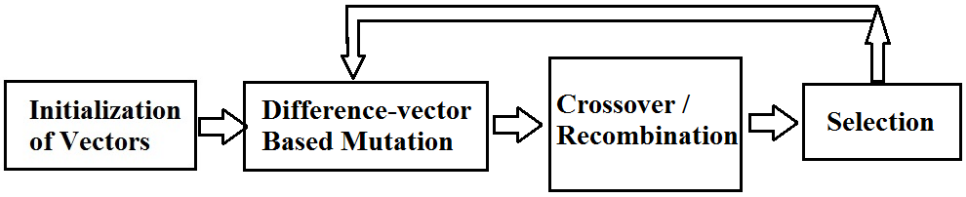

Figure 2-2 Process of Differential Evolution

2.3.1 Initialization of Parameters

The i^{th} vector of the population at the current generation is represented as:

$$X_{i,G} = \left[x_{1,i,G}, x_{2,i,G}, x_{3,i,G}, ..., x_{D,i,G} \right] \tag{4}$$

The subsequent generations in DE are denoted by $G = 0,1,....,G_{max}$. For each parameter of the problem, there may be a certain range within which the value of the parameter should be restricted, therefore the search space is constrained by the prescribed minimum and maximum bounds: $X_{min} = \left\{ x_{1,min}, x_{2,min}, ..., x_{D,min} \right\}$ and $X_{max} = \left\{ x_{1,max}, x_{2,max}, ..., x_{D,max} \right\}$. Hence we may initialize the j^{th} component of the i^{th} vector as in equation (5):

$$x_{j,i,0} = x_{j,min} + rand_{i,j}\left[0,1\right].\left(x_{j,max} - x_{j,min} \right) \tag{5}$$

Where $rand_{i,j}\left[0,1\right]$ is a uniformly distributed random number lying between 0 and 1; and is instantiated independently for each component of the i^{th} vector.

2.3.2 Mutation

Biologically, “mutation” means a sudden change in the gene characteristics of a chromosome. In DE-literature, a parent vector from the current generation is called target vector, a mutant vector

obtained through the differential mutation operation is known as donor vector and finally an offspring formed by recombining the donor with the target vector is called trial vector.

For each target vector $x_{i,G}$, a mutant vector is generated according to equation (6),

$$V_{i,G+1} = x_{r1,G} + F * \left(x_{r2,G} - x_{r3,G} \right) \tag{6}$$

With random indexes $r1, r2, r3 \in \{1, 2, ..., NP\}$, integer, mutually different and $F > 0$. The randomly chosen integers $r1, r2$ and $r3$ are also chosen to be different from the running index i, so that NP must be greater or equal to four to allow for this condition. F is a real and constant factor $\in [0, 2]$ which controls the amplification of the differential variation $\left(x_{r2,G} - x_{r3,G} \right)$.

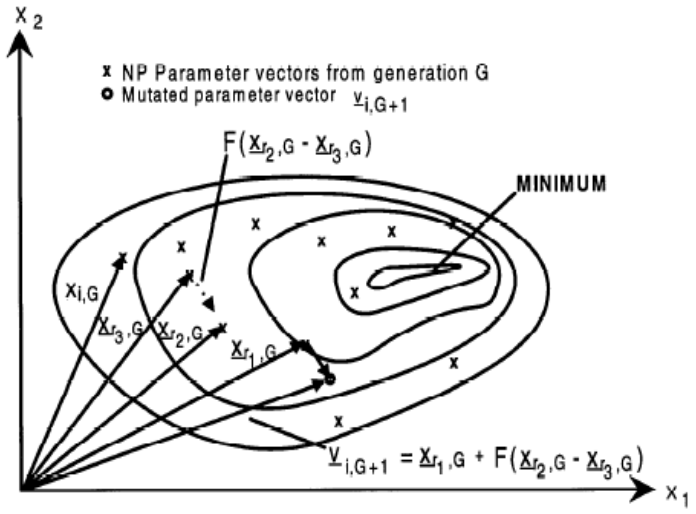

Figure 2-3 A two-dimensional example that illustrates the different vectors which play a part in the generation of $V_{i,G+1}$.

2.3.3 Crossover/Recombination

In order to increase the diversity of the perturbed parameter vectors, crossover is introduced. To this end, the trial vector which is given by equation (7) is formed.

$$u_{i,G+1} = \left[u_{1,i,G+1}, u_{2,i,G+1}, u_{3,i,G+1}, ..., u_{D,i,G+1} \right] \tag{7}$$

Where $u_{j,i,G} = \begin{cases} v_{j,i,G}, & if \quad randb(j) \le Cr \quad or \quad j = rnbr(i) \\ x_{j,i,G}, & if \quad randb(j) > Cr \quad \text{and} \quad j \ne rnbr(i) \end{cases}$

with $j = 1, 2, ..., D$.

$randb(j)$ is the j^{th} evaluation of a uniform random number generator with outcome $\in [0,1]$. Cr is the crossover constant $\in [0,1]$ which has to be determined by the user. $rnbr(i)$ is a randomly chosen index $\in [1,2]$ which ensures that $u_{i,G+1}$ gets at least one parameter from $v_{i,G+1}$. Figure 4 gives an example of the crossover mechanism for 7-dimensional vectors.

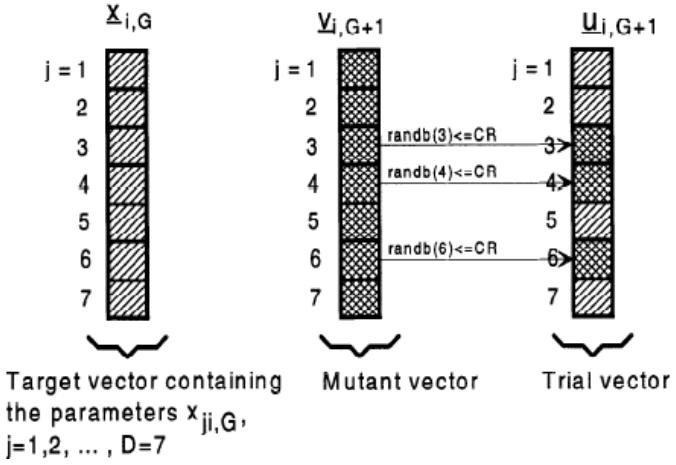

Figure 2-4 Generation of Trial Vector

2.3.4 Selection

To decide whether or not it should become a member of generation $G+1$, the trial vector $u_{i,G+1}$ is compared to the target vector $x_{i,G}$ using the greedy criterion. If vector $u_{i,G+1}$ yields a smaller cost function value than $x_{i,G}$, then $x_{i,G+1}$ is set to $u_{i,G+1}$, otherwise the old value $x_{i,G}$ is retained.

$$X_{i,G+1} = \begin{cases} U_{i,G} & if \;\; f\left(U_{i,G}\right) \le f\left(X_{i,G}\right) \\ X_{i,G} & if \;\; f\left(U_{i,G}\right) > f\left(X_{i,G}\right) \end{cases} \tag{8}$$

2.3.5 Algorithm

Step 1: Read values of the control parameters of *DE* - scale factor F , crossover rate Cr , and the population size *NP* from user.

Step 2: Set the generation number $G = 0$ and randomly initialize a population of *NP* individuals $PG = \left\{X_{1,G}, ..., X_{NP,G}\right\}$ with $X_{1,G} = \left[x_{1,i,G}, x_{2,i,G}, x_{3,i,G}, ..., x_{\mathrm{D},i,G}\right]$ and each individual uniformly distributed in the range $\left[X_{\min}, X_{\max}\right]$, where $X_{\min} = \left\{x_{1,\min}, x_{2,\min}, ..., x_{\mathrm{D},\min}\right\}$ and $X_{\max} = \left\{x_{1,\max}, x_{2,\max}, ..., x_{\mathrm{D},\max}\right\}$.

Step 3: *WHILE* the stopping criterion is not satisfied

DO

FOR $i = 1$ *to NP* //do for each individual sequentially

Step 3.1: Mutation Step- Generate a donor vector $V_{i,G}$ corresponding to the i^{th} target vector $X_{i,G}$ via the differential mutation scheme of *DE* as $V_{i,G} = x_{r1,G} + F * \left(x_{r2,G} - x_{r3,G}\right)$.

Step 3.2: Crossover Step- Generate a trial vector as $u_{i,G+1} = \left[u_{1,i,G+1}, u_{2,i,G+1}, u_{3,i,G+1}, ..., u_{\mathrm{D},i,G+1}\right]$ for the i^{th} target vector $X_{i,G}$ through binomial crossover in the following way:

$$u_{j,i,G} = \begin{cases} v_{j,i,G}, & if \;\; rand_{i,j}\left[0,1\right] \le Cr \;\; or \;\; j = j_{rand} \\ x_{j,i,G}, & otherwise \end{cases}$$

Step 3.3: Selection Step- Evaluate the trial vector $U_{i,G}$.

$$IF \;\; f\left(U_{i,G}\right) \le f\left(X_{i,G}\right), \;\; THEN \;\; X_{i,G+1} = U_{i,G}$$

ELSE $X_{i,G+1} = X_{i,G}$

END IF

END FOR

Step 3.4: Increase the Generation Count

$G = G + 1$

END WHILE

2.3.6 Variants of DE

In order to classify the different variants, the notation: $DE/x/y/z$ is introduced where x specifies the vector to be mutated which currently can be "rand", a randomly chosen population vector, or "best", the vector of lowest cost from the current population; y is the number of difference vectors used, and z denotes the crossover scheme, i.e. exponential (exp) or binomial (bin) [4].

Using this notation, the basic DE-strategy described in the previous sections can be written as: $DE/rand/1/bin$. This is the DE-variant we used for all performance comparisons later on. The other four different mutation schemes, suggested by Storn and Price are summarized as follows:

$$"DE/\text{best}/1": V_{i,G} = X_{best,G} + F.\left(X_{r1,G} - X_{r2,G}\right) \tag{9}$$

$$"DE/\text{target}-\text{to}-\text{best}/1": V_{i,G} = X_{\text{i},G} + F.\left(X_{\text{best},G} - X_{\text{i},G}\right) + F.\left(X_{\text{r1},G} - X_{r2,G}\right) \tag{10}$$

$$"DE/\text{best}/2": V_{i,G} = X_{\text{best},G} + F.\left(X_{\text{r1},G} - X_{\text{r2},G}\right) + F.\left(X_{\text{r3},G} - X_{r4,G}\right) \tag{11}$$

$$"DE/\text{rand}/2": V_{i,G} = X_{\text{r1},G} + F.\left(X_{\text{r2},G} - X_{\text{r3},G}\right) + F.\left(X_{\text{r4},G} - X_{r5,G}\right) \tag{12}$$

$X_{\text{best},G}$ is the best individual vector with the best fitness, i.e. lowest objective function value for a minimization problem, in the population at generation G.

In general, there are a total of ten different working strategies for DE. These strategies are derived from the five different DE mutation schemes outlined above. Each mutation strategy was

combined with either the "exponential" type crossover or the "binomial" type crossover. This yielded a total of 5×2 = 10 DE strategies [4]. In fact many other linear vector combinations can be used for mutation. In general, no single mutation method has turned out to be best for all problems.

2.3.7 Choices of DE's Control Variables

It is interesting to note that DE's control variables, N_P, F and Cr, are not difficult to choose in order to obtain good results.

a) A reasonable choice for N_P is between 5*D and 10*D but N_P must be at least 4 to ensure that DE will have enough mutually different vectors with which to work.

b) As for F, $F = 0.5$ is usually a good initial choice. Values of F smaller than 0.4, like those greater than 1, are only occasionally effective.

c) If the population converges prematurely, then F and/or N_P should be increased.

d) A good first choice for Cr is 0.1, but since a large Cr often speeds convergence, to first try $Cr = 0.9$ or $Cr = 1.0$ is appropriate in order to see if a quick solution is possible.

e) For fastest convergence, it is best to pick the IPR (Initial Parameter Range) such that it covers the region of the suspected global optimum, although this choice doesn't seem to be mandatory.

These rules of thumb for DE's control variables render DE fairly easy to work which is one of DE's major assets.

2.3.8 Advantages of DE

1. Compared to most other EAs, DE is much more simple and straightforward to implement. Main body of the algorithm takes four to five lines to code in any programming language. Simplicity to code is important for practitioners from other fields, since they may not be experts in programming and are looking for an algorithm that can be simply implemented and tuned to solve their domain-specific problems. Note that although PSO is also very easy to code, the performance of DE and its variants is largely better than the PSO variants over a wide variety of problems [4].

2. On non-separable objective functions, the gross performance of DE in terms of accuracy, convergence speed, and robustness still makes it attractive for applications to various real-world optimization problems, where finding an approximate solution in reasonable amount of computational time is much weighted.

3. The number of control parameters in DE is very few (Cr, F and N_P in classical DE). The effects of these parameters on the performance of the algorithm are well studied.

4. The space complexity of DE is low as compared to some of the most competitive real parameter optimizers. This feature helps in extending DE for handling large scale and expensive optimization problems.

The question is why Differential Evolution's results are superior to Genetic Algorithm's. In the Differential Evolution approach each permutation goes through a much more involved type of mutation in the generation of the donor vector than in the mutation step of the Genetic Algorithm. The generation of the trial vector also allows for a greater reach in to the search space by taking individual elements from each permutation than the crossover stage permits with a single break and switch in the Genetic Algorithm.

2.3.9 Drawbacks of DE

1. DE faces significant difficulty on functions that are not linearly separable and can be outperformed by other optimizers. On such functions, DE must rely primarily on its differential mutation procedure, which, unlike its recombination strategy (with $Cr < 1$), is rotationally invariant.

2. DE's mutation strategy lacks sufficient selection pressure when appointing target and donor vectors to have satisfactory exploitative power on non-separable functions. It will be good to use a rank-based parent selection scheme to impose bias on the selection step, so that DE may also learn distribution information from elite individuals in the population and can thus sample the local topology of the fitness landscape better.

3. DE sometimes has a limited ability to move its population large distances across the search space if the population is clustered in a limited portion of it.

4. Some problem landscapes may deceive DE such that it will get stuck in local optima most of the time; however, over similar landscapes PSO will always find the global optima correctly within a maximum time-bound.

5. Also, the performance of DE deteriorates on the spiral "long path problem".

2.3.10 Runtime Complexity

Runtime-complexity analysis of the population-based stochastic search techniques like DE is a critical issue by its own right. In each generation of DE a loop over N_P is conducted, containing a loop over D. Since the mutation and crossover operations are performed at the component level for each DE vector, the number of fundamental operations in $DE/\text{rand}/1/\text{bin}$ is proportional to the total number of loops conducted until the termination of the algorithm. Thus, if the algorithm is stopped after a fixed number of generations $G_{\max}$, then the runtime complexity is $O\left(N_P * D * G_{\max}\right)$.

2.4 IMAGE ENHANCEMENT

2.4.1 Image Enhancement Meaning

Image enhancement is the improvement of digital image in terms of its quality, without any knowledge about the source of degradation. If the source of degradation is known, one calls the process image restoration.

Figure 2-5 General Enhancement Procedure

Image enhancement is basically improving the interpretability or perception of information in images for human viewers and providing `better' input for other automated image processing techniques. The principal objective of image enhancement is to modify attributes of an image to make it more suitable for a given task and a specific observer. During this process, one or more

attributes of the image are modified. The choice of attributes and the way they are modified are specific to a given task. Moreover, observer-specific factors, such as the human visual system and the observer's experience, will introduce a great deal of subjectivity into the choice of image enhancement methods. There exist many techniques that can enhance a digital image without spoiling it. The enhancement methods can broadly be divided in to the following two categories:

a) Spatial Domain Methods

b) Frequency Domain Methods

In spatial domain techniques, we directly deal with the image pixels. The pixel values are manipulated to achieve desired enhancement. In frequency domain methods, the image is first transferred in to frequency domain. It means that, the Fourier Transform of the image is computed first. All the enhancement operations are performed on the Fourier transform of the image and then the Inverse Fourier transform is performed to get the resultant image. These enhancement operations are performed in order to modify the image brightness, contrast or the distribution of the grey levels. As a consequence the pixel value (intensities) of the output image will be modified according to the transformation function applied on the input values.

2.4.2 Commonly Used Image Enhancement Techniques

Many different, often elementary and heuristic methods are used to improve images in some sense. The problem is, of course, not well defined, as there is no objective measure for image quality. These methods are very problem-oriented: a method that works fine in one case may be completely inadequate for another problem. Since spatial domain techniques are used more often than that of frequency domain, some of the common spatial domain based image enhancement techniques are discussed as follows:

a) Point Processing Operation

These are the simplest spatial domain operations like thresholding transformations, log transformation, power law transformations, piecewise linear transformation, contrast stretching.

b) Histogram Processing

This includes manipulation of the histogram of the image. Examples are histogram equalization and histogram matching. Histogram equalization is one of the most common techniques used for enhancement. Histogram equalization is a technique for adjusting image intensities to enhance contrast. This method usually increases the global contrast of many images, especially when the usable data of the image is represented by close contrast values. Through this adjustment, the intensities can be better distributed on the histogram. This allows for areas of lower local contrast to gain a higher contrast as shown in Figure 2-6. Histogram equalization accomplishes this by effectively spreading out the most frequent intensity values.

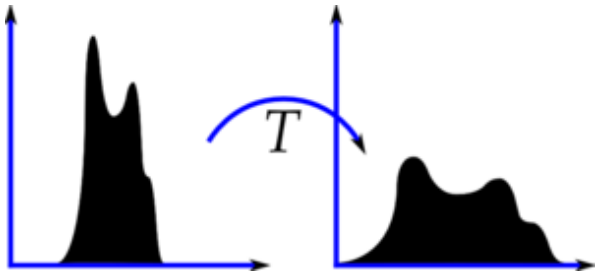

Figure 2-6 Histogram equalization process.

c) Spatial Filtering

This involves an operation on the pixels in a predefined neighborhood about a point; letting the result of that operation be the "response" at that point; and repeating the process for every point in the image. If the computations performed on the pixels of the neighborhoods are linear, the operation is called *linear spatial filtering*; otherwise it is called *nonlinear spatial filtering*.

2.5 LITERATURE SURVEY

Histogram equalization is the most widely used enhancement technique; however it does not work in those cases where brightness preservation is must to avoid artifacts. To preserve the brightness of the original image, many modifications have been suggested in published writings like the bi-histogram equalization (BBHE) [5] and quadrant dynamic histogram equalization (QDHE) [6]. BBHE divides the histogram into two parts based on the input mean brightness and equalizes the two sub histograms independently.

Other commonly used method for contrast enhancement in images is gamma correction or power-law transformation. But the appropriate gamma value needs to be selected according to the conditions in which the image was captured. Wrong selection of gamma may result in under or over enhancement. Thus the sophistication required to select the value of gamma affects capability of this method.

K. Singh et al. proposed Exposure based Sub-Image Histogram Equalization (ESIHE) [7] in which the histogram is partitioned into two according to the exposure level. Then both the parts of the histogram are equalized individually. Also the histogram is clipped above a threshold level to avoid over enhancement. The results are better than the traditional histogram equalization method.

In last decade the concept of exposure has attained special interest. C.H. Lee et al. proposed a passive automatic exposure mechanism [8] as exposure is one of the main factors to successfully take pictures. It applies standard deviation computation of (discrete cosine transform) DCT coefficients to receive the Exposure Value.

One of the most robust methods for image enhancement is image fusion. Image fusion is the process of combining relevant information from two or more images into a single image. The resulting image will be more informative than any of the input images. In [9], an image fusion based approach, called classified virtual exposure image fusion (CVEIF), is proposed for image enhancement. First, several virtual images having different intensity are generated. Then, a classified image fusion method, which blends pixels in distinct luminance classes using different fusion functions, is proposed to produce a fused image in which every image region is well exposed. The success of image fusion depends on the availability of images taken from multiple sensors or images taken under different illumination conditions. The sophisticated requirement for image fusion makes the process applicable rarely.

Numerous attempts have been made to enhance different areas of the image separately catering to the needs of that specific area and to the characteristics local to that area. Another method to tackle uneven illumination is applying three level gamma correction after dividing the image into

dark, bright and medium tone zones on the basis of maximum fuzzy entropy [10]. This method shows improvement over traditional methods of histogram equalization and gamma correction.

Image processing has to deal with many ambiguous situations. Fuzzy set theory is a useful mathematical tool for handling the uncertainty or ambiguity. In [11], using S-function as membership function, the fuzzy region is found by Particle Swarm Optimization (PSO) algorithm based on the maximum fuzzy entropy principle. A new fuzzy entropy is defined which is maximized using PSO to optimize the parameters of the S- function used as membership function. Another image enhancement process using fuzzy logic is presented in [12] by M. Hanmandlu et al. which uses Gaussian membership function to fuzzify the image and then enhance it using sigmoid function. The parameters of the sigmoid function are determined by minimizing the entropy using the modified univariate algorithm.

In [13], A. Khuneta et al. proposed a method for enhancement of dark images or poorly illuminated images. It used the conventional method of gamma correction but the appropriate value of gamma was found out depending on the exposure of the image. In a badly illuminated or dark image, there are number of pixels with less intensity values are much more than the number of pixels with higher intensities. Therefore, in an under exposed image mean pixel intensity and variance of pixel intensity levels are very small. In the same way, for an over exposed image the mean pixel intensity is high and the variance of pixel intensities is low. Based on this, some fuzzy rules were formulated according to which the exposure of the input image was calculated. These fuzzy rules were:

- IF mean is low AND variance is low THEN exposure level low (under-exposed image).
- IF mean is high AND variance is low THEN exposure level high (over-exposed image).
- IF variance is high THEN exposure level medium (adequately exposed image).
- IF mean is medium THEN exposure level medium (adequately exposed image).

The value of exposure was taken in the range [-1, 1] where negative values described under exposed images and positive values described over exposed images. An exposure value close to 0 was considered to be the best value. Therefore the value of gamma was so chosen to push the exposure value of the enhanced image towards zero, according to the following function.

$$\gamma = k^{\lambda} \tag{13}$$

Where γ is gamma and λ is exposure. For dark images λ is between -1 and 0 and γ should lie in [0, 1]. Here k is a positive integer constant whose value is determined by some optimization algorithm so that the exposure of the output image is zero.

Taking just one gamma value and enhancing the whole image using that one gamma value is over simplification of the enhancement problem. Also, the method used for enhancing an under exposed image cannot be applied to an over exposed image. Moreover, in real life, an image with only under exposed area or only over exposed area is rarely found. Generally every image found is of mixed type, containing both under exposed and over exposed regions, which should be dealt with separately.

For enhancement of highly underexposed regions in an image, a simple approach is suggested in [14], using power law transformation. In [15], K. Hasikin presents the fuzzy image enhancement for low contrast and non-uniform illumination images. Again the image is fuzzified using the S-function as membership function. Then the image is modified using power law transformation. For enhancement of under exposed regions gamma is taken as 0.5 and for over exposed regions gamma is taken as 2. In [16], a new fuzzy intensity measure is proposed to distinguish between the dark and bright regions. This measure is computed by considering the average intensity and deviation of the intensity distribution of the image. The input image is enhanced using a power-law transformation.

For underexposed images, [17] presents an enhancement method using parametric sigmoid function. A new objective measure called contrast information factor is introduced which is optimized using PSO to learn the parameters. For images containing underexposed and overexposed regions, both regions are treated separately for fuzzification and enhancement in [18]. The underexposed and the overexposed regions are fuzzified using Gaussian and triangular membership functions and modified using sigmoid and power law function, respectively. The cost function is entropy constrained by visual factor. A similar approach is used in [19] but the image is divided into three parts instead of two according to intensity exposition. F C Cheng et al. [20] propose a new method to enhance the contrast of the input image and video based on Bezier curve. In order to enhance the quality and reduce the processing time, control points of

the mapping curve are automatically calculated by Bezier curve which performs in dark and bright regions separately. Using the fast and accurate histogram modification allows the proposed method to transform the intensity well for both image and video.

Under exposed images can also be enhanced by using the concept of exposure compensation adapted from photography, as proposed by Chen Jui Chung et al in [21]. The proposed method manipulates the reflection rate to obtain an enhanced image. For exposure correction, the zone system and exposure compensation techniques are utilized to correct pixels with insufficient illumination. In real world, photographers use the Zone system to help determine the best exposure setting of one scene. And then they can compensate for the exposure value (EV) to change the setting of camera in order to adjust the shutter speed and aperture size of camera. Using the same zone system, exposure can be found. The input image is first decomposed to extract reflex lightness, and then retinex theory is applied to separate the two components: illumination and reflection. Afterwards, the two components will be enhanced separately, and then combined together to obtain the exposure enhanced results. However, the result of this method depends heavily on the parameter settings. Choosing any other value than the best value produces artifacts and amplifies noise.

CHAPTER 3
PROPOSED METHODOLOGY

3.1 GENERAL DESCRIPTION

The process is begun by calculating the exposure of the image. The exposure of an image gives the quantitative measure of intensity exposition. A dark image or poorly illuminated image has majority pixels with low intensity values, is an under-exposed image whereas a bright image has majority pixels towards the maximum intensity level, known as over-exposed image. In real life, generally a mixed type of image, with both under-exposed and over-exposed regions, is encountered. An entirely under exposed or over exposed image is found rarely. An under-exposed region or an over-exposed regions has the neighborhood pixels, all, either close to the lowest or the highest gray level in the available dynamic range, respectively. And the differences between their gray levels are very low. In both the cases, it is difficult to retrieve the details in the image. Exposure of an image can be calculated as given by equation (14):

$$\text{exposure} = \frac{1}{L}\frac{\sum_{x=1}^{L} p(x)\cdot x}{\sum_{x=1}^{L} p(x)} \tag{14}$$

Where x is the intensity values, L is the maximum intensity level and $p(x)$ gives the histogram of the image.

The value of exposure is normalized i.e., it lies in [0, 1] where an underexposed image has a value near to zero and an overexposed image has a value close to one. It may be note that exposure of a completely dark/image will be equal to zero and that of a completely bright/white image will be equal to one. An image with an exposure around 0.5 is considered to be visually appropriate. Before beginning the enhancement operation, the image is partitioned into underexposed and overexposed regions so that both the regions can be dealt with separately.

For segmenting the image into these two regions, another factor represented by $'a'$ is calculated whose value lies in the range $[0, \mathrm{L}-1]$ where L is the maximum intensity level. Regions with

pixels having intensity values between 0 and a are underexposed whereas those between a and $L-1$ are overexposed. This factor a can be calculated as given by equation (15):

$$a = L \cdot (1 - \text{exposure}) \tag{15}$$

The quality of an image cannot be quantified according to the visual perception of a human since it may be impressionistic. Fuzzy techniques are powerful in the areas of ambiguity. Therefore, fuzzy approach is adopted to deal with uncertainty in images since they can quantify the uncertainty of images by assigning a degree of membership to each pixel. Image processing using fuzzy techniques compromises of 3 stages:

- Fuzzification of the image
- Manipulation of membership values
- Defuzzification of the image

After the image is divided into two parts according to exposure, a modified Gaussian membership function is employed to fuzzify the underexposed regions and it is as given by equation (16):

$$\mu_{X_u}(x) = \exp\left\{-\left[\frac{x_{\max} - (x_{avg} - x)}{\sqrt{2} f_h}\right]^2\right\} \tag{16}$$

Here $\mu_{X_u}(x)$ denotes the degree of membership of a gray level x to the set of underexposed region denoted by μ; $x_{\max}$ and x_{avg} are the maximum intensity level and the average intensity level in the image respectively; and f_h is called fuzzifier, whose value can be found by equation (17):

$$f_h^{\,2} = \frac{1}{2} \frac{\sum_{x=0}^{L-1} (x_{\max} - x)^4 \, p(x)}{\sum_{x=0}^{L-1} (x_{\max} - x)^2 \, p(x)} \tag{17}$$

To fuzzify the overexposed region, a triangular membership function is used. This membership function is given by equation (18).

$$\mu_{X_o}(x) = \begin{cases} 0 & x < a \\ \dfrac{x-a}{L-a} & x \geq a \end{cases} \tag{18}$$

Where $\mu_{X_o}(x)$ denotes the degree of membership of a gray level x to the set of overexposed region denoted by o.

Using these membership functions the spatial domain of the image is converted into the fuzzy domain. The membership functions characterize both the regions separately. After this, different enhancement operators are applied to both the regions. A parametric sigmoid operator is employed for enhancement of underexposed region and is given by equation (19).

$$\mu'_{X_u}(x) = \frac{1}{1+e^{-t\left(\mu_{X_u}(x)-\mu_c\right)}} \tag{19}$$

Where t is the intensification factor and μ_c is the crossover point. $\mu'_{X_u}(x)$ is the modified membership values of the pixels for the underexposed region. The parametric sigmoid function modifies the original membership values of the underexposed region to enhance that region. In order to enhance the overexposed region a power law operator is used. In an overexposed region the intensities of pixels are all near to highest gray level and the difference between the neighborhood pixels is low. Thus the information content is low. After applying the power law transformation the information content is increased by the power of gamma as shown in equation (20).

$$\mu'_{X_o}(x) = K\left(\mu_{X_o}(x) + \varepsilon\right)^{\gamma} \tag{20}$$

Where $\mu'_{X_o}(x)$ is the modified membership values of the pixels for the overexposed region and gamma is the factor by which the membership values are increased. It may be pointed that if a region has all pixels at minimum or maximum intensity value i.e., exposure equal to zero or one, then no enhancement operator can increase the information content by just working on the intensity component and this kind of regions are considered as permanently degraded areas. This issue can be resolved by operating on the saturation component of the image, especially in the case of permanently degraded overexposed region. By manipulating the saturation component,

the hidden details in such a region can be unveiled. Though blindly varying the saturation may affect the color composition badly. Thus the saturation component should be varied carefully.

Entropy gives the measure information in an image. Shannon entropy, equation (21), gives the measure of fuzziness i.e., the degree of uncertainty in the image information in the fuzzy domain.

$$E = \frac{-1}{L \ln 2}\left[\sum_{x=0}^{a-1}\left[\mu'_{X_u}(x)\ln\left(\mu'_{X_u}(x)\right)+\left(1-\mu'_{X_u}(x)\right)\ln\left(1-\mu'_{X_u}(x)\right)\right]\right.$$
$$\left.+\sum_{x=a}^{L-1}\left[\mu'_{X_o}(x)\ln\left(\mu'_{X_o}(x)\right)+\left(1-\mu'_{X_o}(x)\right)\ln\left(1-\mu'_{X_o}(x)\right)\right]\right] \quad (21)$$

Entropy plays a major role and thus need to be optimized. The entropy is being minimized in order to enhance the details. An image with large entropy contains unnecessary information which hides the basic detail. For refinement of the details the entropy needs to be minimized. Also reducing the entropy below a certain limit can dilute the salient features. Thus the entropy is limited to 70% of its original value. The entropy is being pushed towards 0.7 of its original value. Thus the objective function becomes as shown in equation (22):

$$J = \left|E' - 0.7 * \mathrm{E}\right| \quad (22)$$

3.2 PROPOSED ALGORITHM

Step 1: Convert the given image from RGB into HSV model.

Step 2: Determine the histogram $p(x)$ of the image.

Step 3: Find out the value of exposure and a factor of the image using equation (14) and (15) respectively.

Step 4: Compute the value of fuzzifier, f_h using equation (17).

Step 5: Fuzzify the intensity component of the image into underexposed and overexposed regions using the membership functions as given in equations (16) and (18) respectively.

Step 6: Taking the initial values of u_c, t, γ as 0.5, 7, 2 respectively, calculate the modified membership values of pixels for the underexposed and the overexposed sets using equations (19) and (20).

Step 7: Calculate the initial entropy of the image using equation (21).

Step 8: Set the objective function according to the initial entropy. This needs to be minimized using the differential evolution algorithm to determine the optimized values of (u_c, t, γ). For the first iteration this is considered as the best value found (BEST_TILL_NOW) and compared with.

Step 9: Initialize the essential parameters of Differential Algorithm, which are, number of dimension D (which is 3 here), number of population vector N_P, number of iterations to be carried out $ITERMAX$, the mutation factor F (taken as 0.5) and the crossover probability CP (taken as 0.5).

Step 10: Randomize the initial population.

Step 11: Generate a donor vector $V_{i,G}$ corresponding to the i^{th} target vector $X_{i,G}$ via the differential mutation scheme of DE as: $V_{i,G} = x_{r1,G} + F * \left(x_{r2,G} - x_{r3,G} \right)$

Step 12: Generate a trial vector for the i^{th} target vector $X_{i,G}$ through binomial crossover using equation (7).

Step 13: Evaluate the trial vector $U_{i,G}$ i.e., calculate the modified membership values using equationss (19) and (20) for each population vector. Also calculate the new entropy so formed and the corresponding cost of the objective function.

Step 14: Compare the cost of objective function corresponding to each population vector with the BEST_TILL_NOW. If a new minimum is found update the value of BEST_TILL_NOW and also update the population

Step 15: Increase the number of iteration

Sep 16: Repeat the steps 11-15 until either the stopping criteria is not met and the number of iterations < *ITERMAX*

CHAPTER 4
EXPERIMENTAL RESULTS

The following tables show the comparison between the proposed algorithm and the other enhancement algorithm that uses both entropy and visual factor in its objective function and uses Bacterial Foraging Optimization (BFO) Algorithm in place of Differential Evolution for optimization of the objective function. The comparison is performed in the terms of entropy (difference in entropy of enhanced image and original image in percentage), Laplacian mean square error and time complexity. Both the algorithms were tested on the same machine with 1.80 Ghz processor and 4GB RAM.

Table 4-1 Comparison between the enhancement algorithm that uses visual factor in the objective function and BFO for optimization [18]; and the proposed algorithm, in terms of entropy

	Optimal Fuzzy System for image enhancement using Bacterial Foraging			Proposed Methodology		
Image	Original Entropy	Entropy after Enhancement	Difference in entropy (in %)	Original Entropy	Entropy after Enhancement	Difference in entropy (in %)
Hills	0.6592	0.6614	-0.333737864	0.89864	0.62798	30.11884626
Man	0.5769	0.4686	18.77275091	0.8955	0.62915	29.74316025
Cricketer	0.5193	0.4395	15.36683998	0.55536	0.39038	29.70685681
Caugar	0.4705	0.407	13.49628055	0.4702	0.32936	29.9532114
Doctor	0.4069	0.2885	29.09805849	0.44566	0.31199	29.99371718
Face	0.3949	0.4248	-7.571537098	0.41065	0.28767	29.94764398
Scene	0.6385	0.6418	-0.516836335	0.86663	0.60677	29.98511475
Rose	0.4969	0.4713	5.151942041	0.92537	0.64813	29.95990793
Flower	0.6034	0.4609	23.61617501	0.63532	0.44485	29.98016747
Lena	0.3436	0.3342	2.735739232	0.34753	0.28981	16.6086381
Camera	NA	NA	NA	0.42585	0.29843	29.9213338
	Avg. difference (in %) =		9.981567491			28.71987254

From Table 4-1, it can be seen that the average difference in entropy is reduced considerably. The average difference in entropy of the original image and enhanced image was 9.98% earlier

but using the proposed algorithm the average difference increased to 28.72%. Not only the average entropy of all the images is reduced but the entropy of every image is lesser with the proposed algorithm as compared to the enhanced images of the other algorithm. Moreover, it can be seen that in images hills, face, scene, the entropy increases in the other algorithm whereas it decreases in the proposed algorithm.

Table 4-2 Comparison between enhancement algorithm that uses visual factor in the objective function and BFO for optimization [18]; and the proposed algorithm, in terms of Laplacian Mean Square Error

	Optimal Fuzzy System for image enhancement using Bacterial Foraging	**Proposed Methodology**
Image	**Laplacian MSE**	**Laplacian MSE**
Hills	0.0612	0.0226
Man	0.0197	0.0013
Cricketer	0.067	0.0367
Caugar	0.0987	0.0389
Doctor	0.0887	0.0502
Face	0.1588	0.101
Scene	0.0632	0.0526
Flower	0.1563	0.0453
Lena	0.0881	0.0854
Camera	0.0944	0.0577

In the next table, i.e. Table 4-2, the Laplacian mean square error is compared between the two algorithms. For all the cases, the proposed work gives better results (i.e., a lesser value for Laplacian MSE).

The proposed algorithm uses only entropy in its cost function whereas the other algorithm uses visual factor along with the entropy. There are a huge number of calculations involved for computing visual factor. And computing the visual factor for each iteration makes the situation worse. Also the time complexity of Bacterial Foraging Algorithm is higher than Differential Evolution.

Table 4-3 Comparison between the enhancement algorithm that uses visual factor in the objective function and BFO for optimization [18]; and the proposed algorithm, in terms of execution time

	Optimal Fuzzy System for image enhancement using Bacterial Foraging	Proposed Methodology
Image	**Time (in seconds)**	**Time (in seconds)**
Hills	12.947	0.252201
Man	15.835	0.221703
Cricketer	14.083	0.195004
Caugar	13.375	0.197432
Doctor	14.174	0.241766
Face	15.252	0.225371
Scene	13.948	0.212651
Rose	15.594	0.310618
Flower	14.783	0.170799
Lena	14.098	0.190043
Camera	13.481	0.483074

The simplicity of Differential Evolution makes it favorable to use. The cumulative effect of all these factors results in serious time reduction in case of differential evolution which becomes obvious from the above table.

4.1 The input and the output images

1. Cricketer

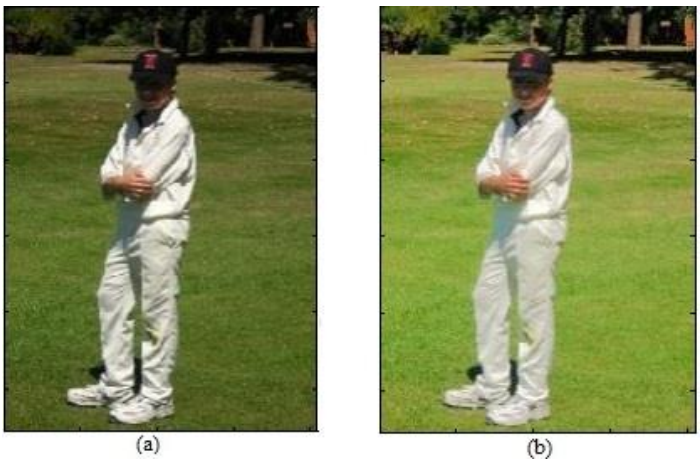

Figure 4-1 (a) original cricketer image with entropy = 0.55536 (b) modified image using proposed technique with entropy = 0.39038

2. Flower

Figure 4-2 (a) original flower image with entropy = 0.63532 (b) modified image using proposed technique with entropy = 0.44485

3. Lena

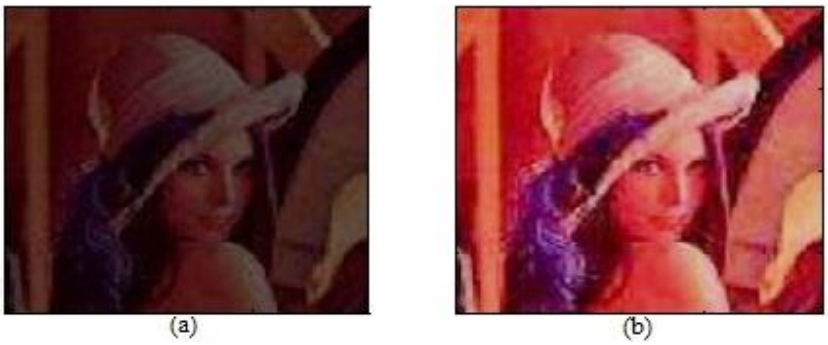

Figure 4-3 (a) original lena image with entropy = 0.34753 (b) modified image using proposed technique with entropy = 0.28981

4. Camera

Figure 4-4 (a) original camera image with entropy = 0.42585 (b) modified image using proposed technique with entropy = 0.29843

5. Hills

Figure 4-5 (a) original hills image with entropy = 0.89684 (b) modified image using proposed technique with entropy = 0.62798

6. Rose

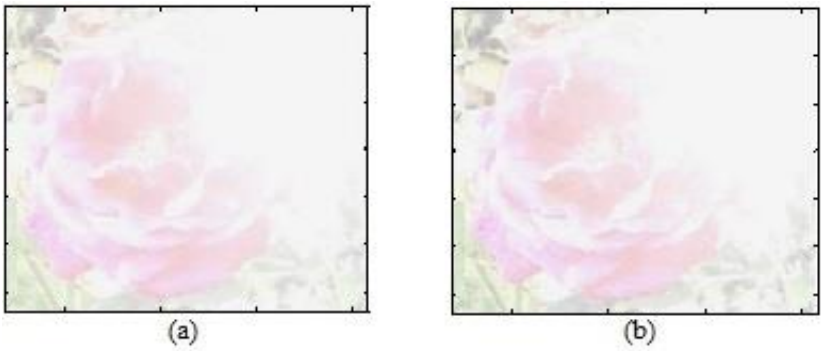

Figure 4-6 (a) original rose image with entropy = 0.92537 (b) modified image using proposed technique with entropy = 0.64813

7. Caugar

Figure 4-7 (a) original caugar image with entropy = 0.4702 (b) modified image using proposed technique with entropy = 0.32936

8. Man

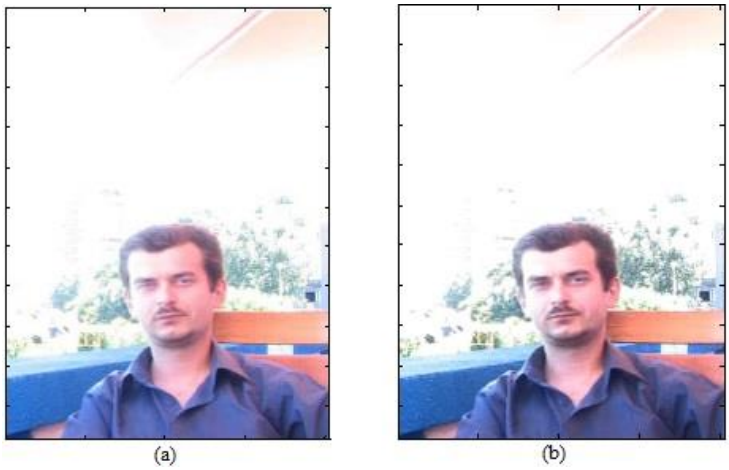

Figure 4-8 (a) original man image with entropy = 0.8955 (b) modified image using proposed technique with entropy = 0.62915

9. Doctor

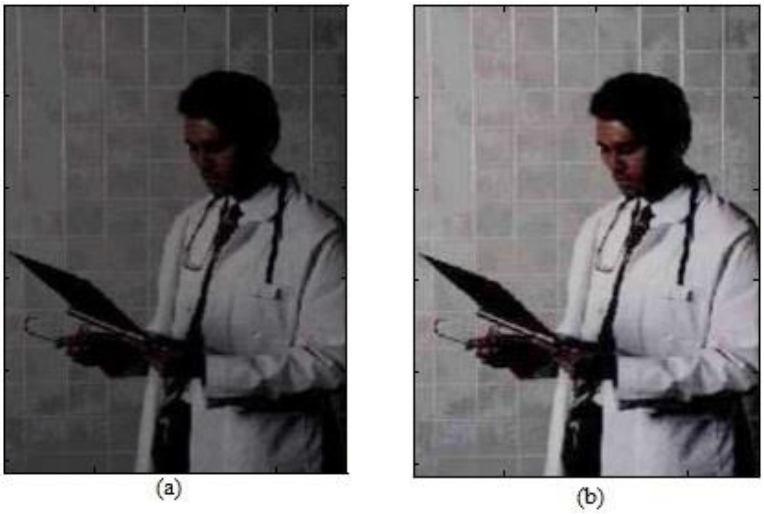

Figure 4-9 (a) original doctor image with entropy = 0.44566 (b) modified image using proposed technique with entropy = 0.31199

10. Face

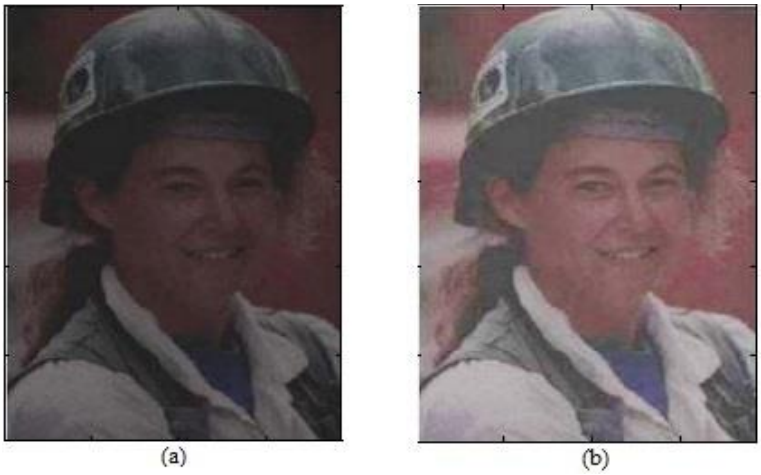

Figure 4-10 (a) original face image with entropy = 0.41065 (b) modified image using proposed technique with entropy = 0.28767

11. Scene

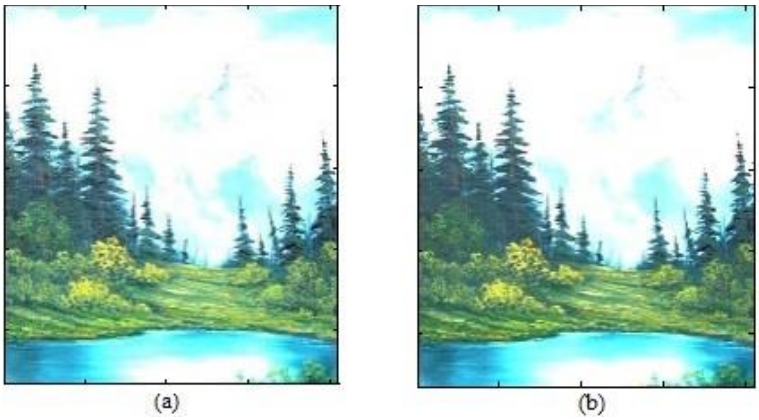

Figure 4-11 (a) original scene image with entropy = 0.86663 (b) modified image using proposed technique with entropy = 0.60677

CHAPTER 5
CONCLUSIONS

Using Differential Evolution over other genetic algorithms proved to be beneficial in terms of time complexity as well as reduction in entropy. DE has emerged as a simple, straightforward and efficient scheme for global optimization. Many versions of DE have been developed to make it a general and fast optimization method for any kind of objective function by twisting and tuning the various constituents of DE, i.e., initialization, mutation, diversity enhancement, and selection of DE as well as by the choice of the control variables. DE outperforms all minimizations approaches in terms of required number of function evaluations necessary to locate a global minimum and that is why there is a major reduction in execution time of the algorithm. DE is also very easy to use as it requires only a few robust control variables which can be drawn from a well-defined numerical interval.

The use of a simple yet meaningful objective function dissolves all the computational complexity of the algorithm given in [19, 20]. By avoiding the calculations involved for computing visual contrast factor, considerable time is saved, hence making it a possible candidate for real time application.

BIBLIOGRAPHY

[1] M. Hanmandlu, O.P. Verma, N.K. Kumar, M. Kulkarni; "A Novel Optimal Fuzzy System for Color Image Enhancement using Bacterial Foraging", IEEE Transactions, Inst. Meas. 58 (8) (2009) 2867–2879.

[2] N. Karaboga, B. Cetinkaya; "Performance Comparison of Genetic and Differential Evolution Algorithms for Digital FIR Filter Design", Advances in Information Systems, Lecture Notes in Computer Science Volume 3261, Page 482-488, Springer- Verlag, Berlin Heidelberg 2005.

[3] R. Storn, K. Price; "Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces", Journal of Global Optimization, Kluwer Academic Publishers, Vol. 11, Pages 341–359, 1997.

[4] S. Das, P. N. Suganthan; "Differential Evolution: A Survey of the State-of-the-Art", IEEE Transactions on Evolutionary Computation, Vol. 15, No. 1, February 2011.

[5] Y.T. Kim, "Contrast Enhancement using Brightness Preserving Bi-histogram Equalization", IEEE Transactions on Consumer Electronics, Vol. 43, No. 1, Page 1–8, February 1997

[6] C. H. Ooi, N. A. M. Isa; "Quadrants Dynamic Histogram Equalization for Contrast Enhancement", IEEE Transactions on Consumer Electronics, Vol. 56, No. 4, Page 2552–2559, November 2010.

[7] K. Singh, R. Kapoor; "Image Enhancement using Exposure based Sub-image Histogram Equalization", Pattern Recognition Letters, Vol. 36, Pages 10–14, Elsevier, 2014.

[8] C. H. Lee, C. M. Huang, J. Y. Liou; "Passive Automatic Exposure Mechanism", Optik, Vol. 124, Pages 4952– 4955, Elsevier, 2013.

[9] Chang-Hsing Lee, Ling-Hwei Chen, and Wei-Kang Wang, "Image Contrast Enhancement Using Classified Virtual Exposure Image Fusion", IEEE Transactions on Consumer Electronics, Vol. 58, Issue 4, pages 1253-1261, 2012

[10] Jinfang Shi, Yong Cai, "A Novel Image Enhancement Method Using Local Gamma Correction with Three-level Thresholding", Information Technology and Artificial Intelligence Conference (ITAIC), 6th IEEE Joint International Conference, Vol. 1, Pages 374 – 378, 2011

[11] SHI Zhen-gang, GAO Li-qun, Wan Kun, "A Novel Approach To Image Enhancement and Thresholding using Fuzzy Theory", Second IEEE Conference on Industrial Electronics and Application, pages 2201-2205, 2007

[12] Madasu Hanmandlu, Devendra Jha, "An Optimal Fuzzy System for Color Image Enhancement", IEEE Transactions on Image Processing, Vol. 15, No. 10, pages 2956-2966, October 2006

[13] A. Khunteta, D. Ghosh, Ribhu; "Fuzzy Rule-based Image Exposure Level Estimation and Adaptive Gamma Correction for Contrast Enhancement in Dark Images", ICSP2012 Proceedings, IEEE, 2012.

[14] Om Prakash Verma, Nitesh Gil, Pooja Gupta, Megha, "A Simple Approach for Image Enhancement using New Power-Law Transformation Operators", IEEE International Conference on Signal Processing and Communication (ICSC), pages 276-281, 2013

[15] Khairunnisa Hasikin, Nor Ashidi Mat Isa, "Enhancement of the low contrast image using fuzzy set theory", 14th International Conference on Modelling and Simulation, IEEE, Pages 371-376, 2012

[16] Khairunnisa Hasikin, Nor Ashidi Mat Isa," Fuzzy Image Enhancement for Low Contrast and Non-uniform Illumination Images", IEEE International Conference on Signal and Image Processing Applications (ICSIPA), pages 275- 280, 2013

[17] M. Hanmandlu, Shaveta Arora, Gaurav Gupta, Latika Singh, "A Novel Optimal Fuzzy Color Image Enhancement using Particle Swarm Optimization", Sixth International Conference On Contemporary Computing (IC3), pages 41-46, IEEE, 2013

[18] Madasu Hanmandlu, Om Prakash Verma, Nukala Krishna Kumar, Muralidhar Kulkarni , "A Novel Optimal Fuzzy System for Color Image Enhancement using Bacterial Foraging", IEEE Transactions on Instrumentation and Measurement, Vol. 58, No. 8, pages 2867- 2879, August 2009

[19] O.P. Verma, Puneet Kumar, Madasu Hanmandlu, Sidharth Chhabra, "High dynamic range optimal fuzzy color image enhancement using Artificial Ant Colony System", Applied Soft Computing Journal, Elsevier, 2011

[20] Fan-Chieh Cheng, Shih-Chia Huang, "Efficient Histogram Modification Using Bilateral Bezier Curve for the Contrast Enhancement", Journal of Display Technology, Vol. 9, Issue 1, pages 44-50, IEEE, 2013

[21] C. J. Chung, W. Y. Chou, C. W. Lin; "Under-exposed Image Enhancement using Exposure Compensation", 13th International Conference on ITS Telecommunications (ITST), 2013.

[22] Sk. Minhazul Islam, Swagatam Das, P. N.Suganthan- "An Adaptive Differential Evolution Algorithm with Novel Mutation and Crossover Strategies for Global Numerical Optimization", IEEE Transactions on Systems, Man, and Cybernetics, Vol. 42, Issue 2, pages 482-500, April 2012

[23] Swagatam Das, A. Biswas, S. Dasgupta , and A. Abraham, "Bacterial Foraging Optimization Algorithm: Theoretical Foundations, Analysis, and Applications", Foundations of Comp. Intelligence, Vol 3, pages 23-55, Springer- Verlag, Berlin Heidelberg 2009

[24] Kevin M. Passino, "Biomimicry of Bacterial Foraging for Distributed Optimization and Control, IEEE Control Systems Magazine", June 2002