AN IMAGE ENHANCEMENT ALGORITHM USING GAMMA CORRECTION BY SWARM OPTIMIZATION

Farah Shabana[1], Sreekanth Badithala[2], Saibabu Daggupati[3], Rambabu Chevala[4], Dr. Kalyan Raj Kaniganti[5]

[1,2,3]IV B.Tech students, Department of ECE, [4]Senior Grade Assistant Professor, Department of ECE, [5]Associate Professor, Department of EEE, Gudlavalleru Engineering College, Gudlavalleru, Krishna District, A.P, India.

---***---

Abstract-*The gist of this paper is complete examination identified after analysing existing optimization techniques with the proposed technique. The central goal of this work is to increase the information content and enhance the details of an image using an adaptive gamma correction technique aided by swarm optimization. Gamma correction is a well established technique that preserves the mean brightness of an image that produces natural looking images by the choice of an optimal gamma value. Here, the basic optimization that is particle swarm optimization which is employed to estimate an optimal gamma value. In the proposed method, we obtain an optimized gamma value for an image and an optimized solution for our problem within less time. The proposed method is compared with the Linear Contrast Stretching (LCS), Histogram Equalization (HE) in terms of iteration number. Simulation results demonstrate that the proposed swarm optimization based contrast enhancement method improves the overall image contrast and enriches the information present in the image within less time.*

Keywords*- Constrained Optimization, Linear Contrast Stretching (LCS), Histogram Equalization (HE), Particle Swarm Optimization (PSO)*

1. INTRODUCTION

An image is an array or matrix, of square pixels (picture elements) arranged in columns and rows. An image is defined on, say a photographic film, is a continuous function of brightness values. An image is nothing more than a 2-dimensional signal. It is defined by the mathematical function f(x, y) where x and y are the two co-ordinates horizontally and vertically. The value of f(x, y) at any point gives the pixel value at that point of an image. At each pixel (or at each grid square) we usually represent the gray level value using an integer ranging from 0 for black to 255 for fully white.

Image processing is a method to perform some operations on an image, in order to get an enhanced image or to extract some useful information from it. It is a type of signal processing in which input is an image and output may be image or characteristics/features associated with that image. Now-a-days, image processing is among rapidly growing technologies. It forms core research area within engineering and computer science disciplines too.

It basically includes the following three steps:

a. Importing the image via image acquisition tools.
b. Analysing and manipulating the image;
c. Output in which the result can be altered image or report that is based on image analysis.

Image processing is often viewed as arbitrarily manipulating an image to achieve an aesthetic standard or to support a preferred reality. However, image processing is more accurately defined as a means of translation between the human visual system and digital imaging devices. The human visual system does not perceive the world in the same manner as digital detectors, with display devices imposing additional noise and bandwidth restrictions. Salient differences between human and digital detectors will be shown, along with some basic processing steps for achieving translation. Image processing must be approached in a manner consistent with the scientific method so that others may reproduce, and validate, one's results. This includes recording and reporting processing actions, and applying similar treatments to adequate control images.

The stages of digital image processing are shown briefly in figure 1

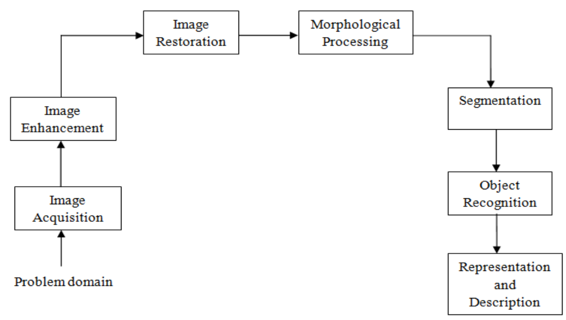

Figure 1: Stages of Digital Image Processing

1.1 Image Enhancement

The aim of image enhancement is to improve the interpretability or perception of information in images for human viewers, or to provide 'better' input for other automated image processing techniques.

Image enhancement techniques can be divided into two broad categories:

A. Spatial domain:

This operates directly on pixels.

B. Frequency domain:

This operates on the fourier transform of an image.

I. Spatial domain enhancement methods

Spatial domain techniques are performed to the image plane itself and they are based on direct manipulation of pixels in an image. The operation can be formulated as g(x,y) = **T**[f(x,y)],
where g is the output, f is the input image and **T** is an operation on f defined over some neighborhood of (x,y).

A. Spatial Filtering

The use of spatial masks for image processing is called spatial filtering. The masks used are called spatial filters.

B. Smoothing filter

Smoothing filters are used for blurring and for noise reduction. Blurring is used in preprocessing steps, such as removal of small details from an image prior to object extraction, and bridging of small gaps in lines or curves. Noise reduction can be accomplishing by blurring with a linear filter and also by nonlinear filtering.

C. Sharpening Filters

To highlight fine detail in an image or to enhance detail that has been blurred, either in error or as a natural effect of a particular method of image acquisition. Uses of image sharpening vary and include applications ranging from electronic printing and medical imaging to industrial inspection and autonomous target detection in smart weapons.

D. Median filtering

If the objective is to achieve noise reduction instead of blurring, this method should be used. This method is particularly effective when the noise pattern consists of strong, spike-like components and the characteristic to be preserved is edge sharpness. It is a nonlinear operation. For each input pixel f(x,y), we sort the values of the pixel and its neighbors to determine their median and assign its value to output pixel g(x,y).

E. Derivative filters

Differentiation can be expected to have the opposite effect of averaging, which tends to blur detail in an image, and thus sharpen an image and be able to detect edges. The most common method of differentiation in image processing applications is the gradient. For a function f(x,y), the gradient of f at coordinates (x',y') is defined as the vector. Its magnitude can be approximated in a number of ways, which result in a number of operators such as Roberts, Prewitt and Sobel operators for computing its value.

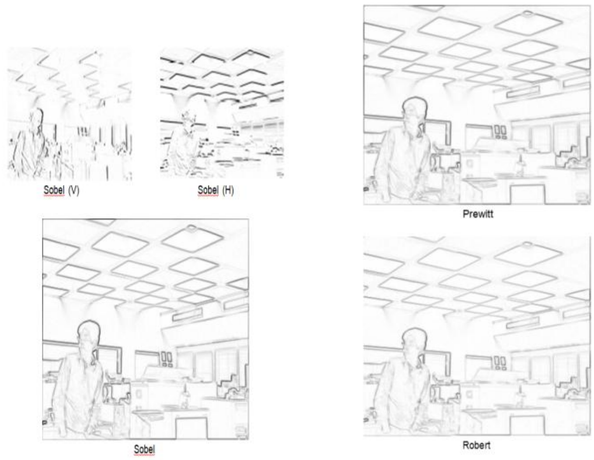

Figure 2: Edge detection with various operators (Outputs are normalized.)

II. Frequency domain enhancement methods

These methods enhance an image f(x,y) by convoluting the image with a linear, position invariant operator. The 2D convolution is performed in frequency domain with DFT.

Spatial domain: g(x,y)=f(x,y)*h(x,y)
Frequency domain: G(w1,w2)=F(w1,w2)H(w1,w2)

We simply compute the Fourier transform of the image to be enhanced, multiply the result by a filter transfer function, and take the inverse transform to produce the enhanced image.

Spatial domain: g(x,y)=f(x,y)*h(x,y)
Frequency domain: G(w1,w2)=F(w1,w2)H(w1,w2)

A. Low pass filtering

Edges and sharp transitions in the gray levels contribute to the high frequency content of its Fourier transform, so a low pass filter smoothes an image.

Formula of ideal LPF

$$H(u,v)= \begin{cases} 1 & \text{if } D(u,v)<=D_0 \\ 0 & \text{else} \end{cases}$$

The plot of fourier transform of ideal and butterworth LPF is shown in fig 3

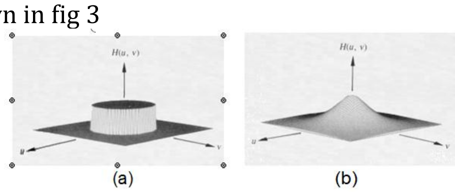

Fig 3: (a) Ideal LPF, (b) Butterworth LPF

B. High pass filtering

A high pass filter attenuates the low frequency components without disturbing the high frequency information in the Fourier transform domain can sharpen edges.
Formula of ideal HPF function

$$H(u,v)=\begin{cases} 0 & \text{if } D(u,v)<=D_0 \\ 1 & \text{else} \end{cases}$$

The plot of fourier transform of ideal and butterworth HPF is shown in fig 4.

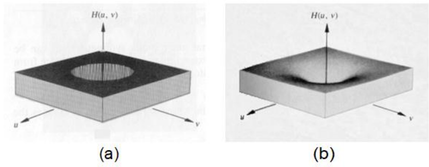

Fig 4: (a) Ideal HPF, (b) Butterworth HPF

2. LITERATURE REVIEW

With the explosion of data generation, getting optimal solutions to data driven problems is increasingly becoming a challenge, if not impossible. It is increasingly being recognised that applications of intelligent bio-inspired algorithms are necessary for addressing highly complex problems to provide working solutions in time, especially with dynamic problem definitions, fluctuations in constraints, incomplete or imperfect information and limited computation capacity. More and more such intelligent algorithms are thus being explored for solving different complex problems while some studies are exploring the application of these algorithms in a novel context, other studies are incrementally improving the algorithm itself. However, the fast growth in the domain makes researchers unaware of the progresses across different approaches and hence awareness across algorithms is increasingly reducing, due to which the literature on bio-inspired computing is skewed towards few algorithms only like genetic algorithm, particle swarm algorithm etc.

At times the images captured from various sources lack enough contrast and look washed out. Although they look bright enough, they still look like something is wrong with them.

In optimization, we start with some kind of initial values for the variables used in the experiment. Because these values may not be the best one to use, we should change them until we get the best ones. In some cases, these values are generated by complex functions that we cannot solve manually. But it is very important to do optimization because a classifier may produce bad classification accuracy not because, for example the data is noisy or the used learning algorithm is weak but due to the bad selection of learning parameters initial values. As a result there are different optimization techniques suggested by the operation research (OR) researchers to do such work of optimization.

Optimization techniques are categorized into four categories:

a. Constrained optimization
b. Multimodal optimization
c. Multi objective optimization
d. Combinational optimization

a. Constrained optimization

In mathematicaloptimization, constrained optimization is the process of optimizing an objective function with respect to some variables in the presence of constraints on those variables. The objective function is either a cost function or energy function, which is to be minimized, or a reward function or utility function, which is to be maximized. Constraints can be either hard constraints, which set conditions for the variables that are required to be satisfied, or soft constraints, which have some variable values that are penalized in the objective function if, and based on the extent that, the conditions on the variables are not satisfied.

General forms

A general constrained minimization problem may be written as follows:

Min f(x)

Subject to $g_i(x)= c_i$ for i= 1,.....,n Equality constraints

$h_j(x)>= d_j$ for j=1,.....,m Inequality constraints

where $g_i(x) = c_i$ for i= 1,...,n and $h_j(x)>= d_j$ for j=1,.....,m are constraints that are required to be satisfied and f(x) is the objective function that needs to be optimized subject to the constraints.

2.1 Evolutionary algorithms

We can say that optimization is performed using evolutionary algorithms (EA's). The difference between traditional algorithms and EA's is that EAs are not static but dynamic as they evolve over time.

Evolutionary algorithms have three main characteristics:

a. Population based

The set of current solutions from which new solutions are to be generated is called the population.

b. Fitness-oriented

There is a fitness value associated with each other individual solution calculated from a fitness function. Such fitness values represent how good the solution is.

c. Variation-driven

If there is no acceptable solution in the current population according to the fitness function calculated from each individual, we should make something to generate new better solutions. As a result, individual solutions will undergo a number of variations to generate new solutions.

Images being bright has to do with the amplitude of their signal while them having low contrast is a fault in the signal's frequency. The above diagram shows example of low contrast image. Now there is a need to mathematically process the signals and fix them. The first and foremost simple technique is called Linear Contrast Stretching.

2.2 Linear Contrast Stretching

Contrast is the difference between maximum and minimum pixel intensity. This technique improves an image by stretching the range of intensity values.

The formula for performing this stretching is as follows.

R(x, y) = (I(x, y) – Imin) * ((Lmax – Lmin)/(Imax – Imin)) + Lmin

I(x, y) →input image defined as function of coordinates

Imax, Imin→maximum and minimum values of the image

Lmax, Lmin→maximum and minimum signal value range (typically are 255,0)

Although the technique is simple and quite useful to implement, it is vulnerable to outlier pixel values. Suppose the image has all the pixels in the range [200-255] except one which is completely black i.e has a value of 0, then the contrast stretching won't work at all.

A more robust technique is Histogram Equalization

2.3 Histogram Equalization

Histogram is a data-structure to store the frequencies of all pixel levels in the images, which simply mean the number of pixels in the image which have that specific pixel intensity value. The number of bins for the histogram in this case are taken to be equal to the number of pixel intensity levels in the image. Usually, it is 0-255.

For equalizing histogram, we need to compute the histogram and then normalize it into a probability distribution. For normalization, we just need to divide the frequency of each pixel intensity value by the total number of pixels present in the image. This is equal to the resolution of the image i.e. number of rows x number of columns. The equalization process makes sure that the resulting histogram is flat. Following is the transformation function for the image in order to obtain a flat histogram.

T(x, y) = (L-1) * (sum(k,{0-P(I(x, y))}, Pk(I(x, y))))

Sum(k, {a,b}, fk(x))→summation over f(x) where k goes from a to b

In order to understand the above formula, we need to know the concept of Cummulative Mass function (CMF). The CMF of a random variable can take upto the given value. So, basically in the Histogram Equalization formula we are transforming the value of input pixel intensity to such a value which is in sync with the CMF of the input image.

Flow diagram of Histogram equalization:

The above detailed explanation of histogram equalization is drawn using a flow diagram which is shown in fig 5.

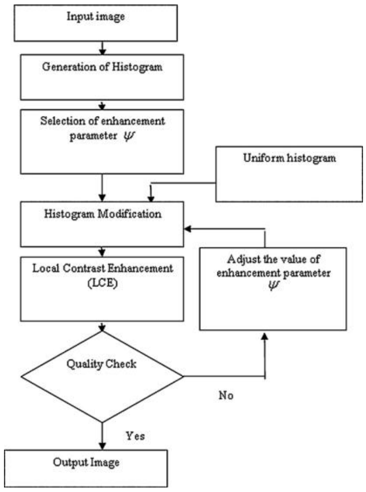

Fig 5: Flow diagram of Histogram Equalization

The comparative analysis of a normal image with its histogram equalized image along with their respective histograms in both the cases is shown in fig 6.

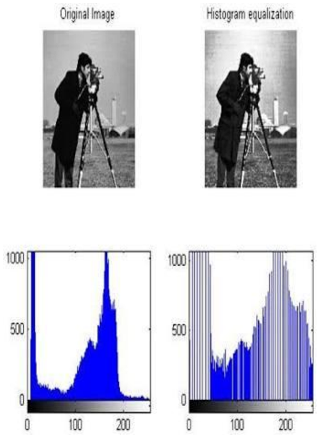

Fig 6: Comparison of original and Histogram Images

This method is indiscriminate. It may increase the contrast of the background noise, while decreasing the usable signal. It also fails when the input image has large area and low intensity background. In this case, the histogram has a spike component corresponding to the background gray level. After histogram equalisation, the output image has a severe washed-out appearance while its dynamic range actually becomes smaller. It is effective only if the original image has poor contrast to start with. Otherwise it may degrade the image quality.

3. DESCRITPTION OF SWARM OPTIMIZATION

Optimization problem is a computational problem in which the object is to find the best of all possible solutions. In other word, optimization problem is to find a solution in the feasible region which has the minimum (or maximum) value of the objective function.

Single objective optimization is often used by researchers in solving real world problem.

Sometimes, a better way will be achieved by defining multiple objectives in solving a problem.

Multi-objective optimization is a process of solving a problem by simultaneously optimizing two or more objectives subjected to constraints. Optimal solution is solution that is not dominated by other solution in the search space where the optimal solution is called Pareto optimal.

3.1 Metaheuristic Algorithm

Metaheuristic is an iterative generation process which guides a subordinate heuristic by combining intelligently different concepts for exploring and exploiting the search space while learning strategies are used to structure information to find efficiently near-optimal solutions.

Concluded that metaheuristic approaches are very suitable for solving non-deterministic.

Polynomial hard (NP-hard) optimization and complicated search problems because they obtained better quality solutions compare to heuristic approaches especially on hybrid techniques.

Furthermore, the approaches are more efficient to solve the problems.

This is proved by, metaheuristics are designed to deal with complex optimization problems because of the other optimization techniques are not very efficient to solve the problems. Hence, these approaches were recognized as one of the most practical approaches for solving complex problems.

3.2 Fundamental Properties of Metaheuristics

- Metaheuristics are strategies that act as guidance for searching process.
- The goal of metahueristics is to explore the search space efficiently to find optimal or near-optimal solutions.
- The techniques used to solve problems are from simple local search procedures to complex learning process.
- Metaheuristic algorithms are approximation solutions and usually non-deterministic.

The most significant algorithms that have many contributions to the field are Genetic Algorithm (GA), Firefly Algorithm (FA), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), Levy Flight, Artificial Bee Colony Algorithm (ABC), Hunting Search (HuS), Simulated Annealing (SA) and many more.

3.3 Gamma Correction

Gamma correction is a non-linear adjustment to individual pixel values. While in image normalization we carried out linear operations on individual pixels, such as scalar multiplication and addition/subtraction, gamma correction carries out a non-linear operation on the source image pixels, and can cause saturation of the image being altered. Furthermore, it can also lead to poor contrast if the gamma value is too large or too small.

At first glance, gamma correction appears to either darken or brighten an image, but this is a gross oversimplification. We can already adjust the average brightness of an image by some modified normalization algorithm, the simplest of which would be simply adding a constant value to each pixel intensity value, effectively “shifting” the mean pixel intensity values across the entire image. Gamma correction is an operation that effectively carries out an exponential function on individual pixel values.

In the case of the ideal display of television device below, the desired output intensity value is shown in fig 7.

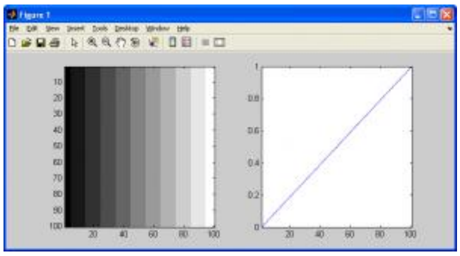

Fig 7: Normal Intensity Value Gamma=1.0

Now consider an old television set that is on its last legs. Due to worn out components, everything appears darker than it should, as demonstrated in fig 8.

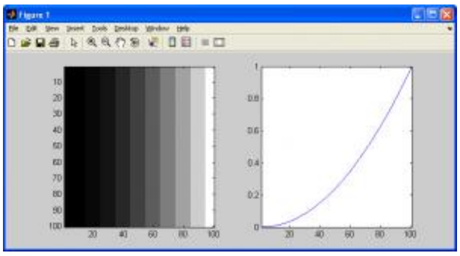

Fig 8: Dark Image Gamma=2.0

Finally, consider the case of a new TV that is far too bright. The Output spectrum and transfer function are shown in fig 9.

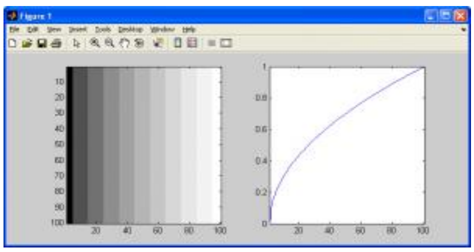

Fig 9: Brighter Image Gamma=0.5

A table of figures representing the transfer function of gamma correction for varying values of gamma is shown in fig 10 for emphasis.

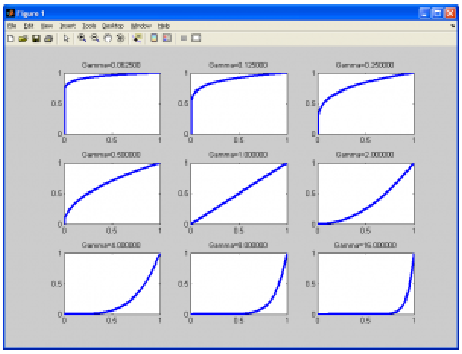

Fig 10: Varying gamma values

In the ideal case, with $\gamma = 1$, our input and output intensity values match perfectly, as in the case of the ideal display described at the top of the page. As the gamma value deviates from unity, we begin to notice a significant difference in the input and output intensity values. As gamma approaches zero, the output pixels become brighter, while as gamma approaches infinity, the pixels become darker. Gamma correction essentially carries out the inverse function of the exponential operation caused by imperfections in the display device.

Comparision of a simple image with varying gamma values is shown in fig 11.

Fig 11: Simple image with varying gamma values

3.4 Particle swarm optimisation

Particles Swarm Optimization (PSO) is an evolutionary computation technique originally developed by Kennedy and Eberhart (1995). The PSO is motivated from the stimulation of social behavior instead of evolution of nature as in the other evolutionary algorithms (genetic algorithms, evolutionary programming, evolutionary strategies, and genetic programming). PSO is sociologically inspired, since the algorithm is based on sociological behavior associated with bird flocking. It is a population based evolutionary algorithm. Similar to the other population based evolutionary algorithms, PSO is initialized with a population of random solutions.

The algorithm maintains a population of particles, where each particle represents a potential solution to an optimization problem Unlike the most of the evolutionary algorithms, each potential solution (individual) in PSO is also associated with a randomized velocity, and the potential solutions called particles, are then flown through the problem space.

Let S be the size of the swarm, each particle i can be represented as an object with several characteristics. A population of particles is initialized with random position Xi and velocities Vi objective function Fi is evaluated using the particles positional coordinates as input values. Each particle keeps track of its coordinates in the problem space, which are associated with the best solution (fitness) it has achieved so far. This value is called p_{best}. Another best value that is tracked by the global version of the swarm is the overall best value, and its location obtained so far by any particle in the population. This location is called gbest. At each time step velocity of each particle flying toward its gbest and pbest location is changed. Acceleration is weighted by random terms, with separate random numbers being generated for acceleration towards pbest and gbest location.

At each time step position and velocities are adjusted and the function is evaluated with new coordinates. When the particle discovers a pattern that is better than any it has found previously, it stores the coordinates in the vector p_{bestid}. The difference between the best point found by a particular agent and the individual's current positions is stochastically added to the current velocity causing the trajectory to oscillate around the point. Further each particle is defined within the context of a topological neighborhood comprising itself and some other particles in the population. The stochastically weighted difference between the neighborhood's best position gbestd and the individual's current position is also added to its velocity, adjusting it for the next time step. These adjustments to the particle's movement through the space cause it to search around the two best positions.

3.5 PSO Algorithm

PSO algorithm consists of the following steps:

Step 1: Initialization:
Initialize a population of particles with random position and velocities in d dimensional problem space. Confine the search space by specifying the lower and upper limits of each decision variable. The populations of points are initialized with the velocity and position set to fall into the pre-specified or allowed range and satisfying the equality and inequality constraints.

- Evaluate the fitness of each particle in terms of paretodominance.
- Record the non dominated solutions found so far and save them in archive.
- Initialize the memory of each individual where the personal best position is stored.
- Choose the global best position $g_{best}^{(t)}$ from the archive. Increase the generation number.

Step 2: Velocity updating:
At each iteration, the velocities of all particles are updated which is:

$V_{id}^{(t+1)}=wV_{id}^{(t)}+c_1rand_1(p_{bestid}^{(t)}-X_{ic}^{(t)})+c_2rand_2(g_{bestid}^{(t)}X_{id}^{(t)})$

where $V_{id}^{(t)}$ and $X_{id}^{(t)}$ are the velocity and position of particle i, in d dimensional space respectively. $p_{bestid}^{(t)}$ is the best position of individual i in d dimensional space until generation t; $g_{bestd}^{(t)}$ is the best position of the group in d dimension until generation t; w is the inertia weight factor controlling the dynamics of flying; c1 and c2 are accelerating constants; rand1 and rand2 are random variables in the range [0,1].

The population size selected is problem dependent. Population size of 20-50 is most common. Under the multi–objective environment the number of non dominated solutions is directly linked to the population size. So a larger population is preferred.

Step 3: Position updating:
Between successive iterations, the position of all particles are updated which is

$$X_{id}^{(t+1)}=X_{id}^{(t)}+V_{id}^{(t+1)}$$

where $X_{id}^{(t+1)}$ is the new position and $X_{id}^{(t)}$ is the previous position and $V_{id}^{(t+1)}$ is the new velocity.

Step 4: Memory updating:
Update particle best position $p_{best}^{(t)}$ and global best position $g_{bestd}^{(t)}$ using below equation.

$$P_{bestid}^{(t+1)} \leftarrow X_{id}^{(t+1)} \text{ if } f(X_{id}^{(t+1)} < f(p_{bestid}^{(t)})$$

$$g_{bestd}^{(t+1)} \leftarrow X_{id}^{(t+1)} \text{ if } f(X_{id}^{(t+1)} < f(g_{bestd}^{(t)})$$

where f(X) is the objective function to be minimized. Compare particles fitness evaluation with particles $p_{bestid}^{(t)}$. If current value is better than $p_{bestid}^{(t)}$ then set $p_{bestid}^{(t+1)}$ value equal to the current value and the $p_{bestid}^{(t+1)}$ location equal to the current location in d dimensional space. Compare fitness evaluation with the population's overall previous best. If the current value is better than $g_{bestd}^{(t)}$ then reset $g_{bestd}^{(t+1)}$ to the current particles array index and value.

Step 5: Termination criteria examination:
The algorithm repeats Step 2 to Step 4 until a sufficient good fitness or a maximum number of iterations/epochs are reached. Once terminated, the algorithm outputs the points of $g_{best}^{(t)}$ and $f(g_{bestd}^{(t)})$ as its solution.

Optimal Parameters of PSO: For all the problems considered the following parameters give optimal results:
Population size = 50
Number of iterations = 100 c1, c2 = 2
w is varied from 1.4 to 0.4.
Maximum velocity Vmax is limited to 10% of the dynamic range of the variables on each dimension.

3.6 PSO Flow diagram:

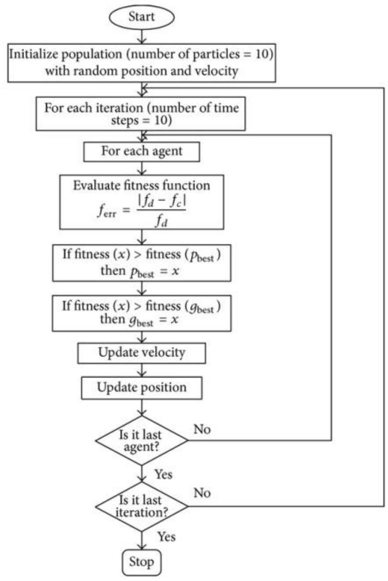

Fig 12: Flow diagram of PSO algorithm

3.7 PSO algorithm:

Read Input image (size of M × N)
Initialize dimension, number of particles, position, velocity, and iterations (t)
for each iteration
for each particle
for each dimension
Apply gamma correction to input image
Compute entropy of enhanced image
Compute edge content of the enhanced image
Evaluate the fitness function and store in temp
if (temp>gbest_val)
Store current particle location value as global best end
if (t≠ maximum_no_iterations)
Update velocity
Update position
Repeat line number 6 to 15 until the termination criterion

End
else
save gbest as optimal gamma value
end
end
end
end

Comparative analysis done to a normal image with a PSO optimized image is shown in the fig 13.

Fig 13: Original image and a PSO optimized image

4. RESULTS DESCRIPTION

Data set1: Analysis of an image captured by a mobile phone:

Here, in this case, the picture clarity wont be upto the mark and hence, more no: of iterations are needed to obtain an optimized image.

The original image is shown in fig 14.

The contrast histogram equalized image is shown in fig 15.

The proposed optimized image is shown in the fig 16.

Fig 14: Original Image

Fig 15: Histogram Equalized Image

Fig 16: Proposed Optimized Image

The no: of iterations needed to optimize this data set is 134.

The performance metrics i.e., average pixel intensity, standard deviation and entropy of the above camera picture, its histogram equalized and its proposed optimized image are given in the tabular form 1.

Table 1: Image quality metrics of Data set-1

	Gamma Value	Average Pixel Intensity	Standard Deviation	Entropy
Original Image	0.5	106.22	64.60	7.68
Histogram Equalized Image	0.5	139.12	73.51	7.82
Proposed Optimized Image	6.5	111.96	92.67	5.59

Data set2: Analysis of a High Resolution image:

Here, in this case, the picture clarity is extraordinary and hence, very less no: of iterations are needed to obtain an optimized image.

The original image is shown in fig 17.
The contrast histogram equalized image is shown in fig 18.
The proposed optimized image is shown in the fig 19.

Fig 17: Original Image

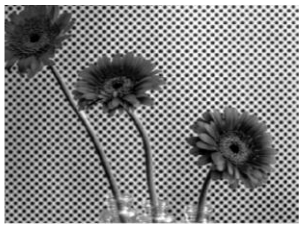

Fig 18: Histogram Equalized Image

Fig 19: Proposed Optimized Image

The no: of iterations needed to optimize this data set is 42. The performance metrics i.e., average pixel intensity, standard deviation and entropy of the above High resolution image, its histogram equalized and its proposed optimized image are given in the tabular form 2.

Table 2: Image quality metrics of Data set-2

	Gamma Value	Average Pixel Intensity	Standard Deviation	Entropy
Original Image	0.5	149.9	55.0	7. 52

Histogram Equalized Image	0.5	172.9	63.34	7.45
Proposed Optimized Image	1.11	100.55	99.24	5.4

5. ADVANTAGES OF PSO

- A PSO is considered as one of the most powerful methods' for resolving the non-smooth global optimization problems.
- It has many key advantages as follows: PSO is a derivative free technique just like as other heuristic optimization techniques.
- PSO is easy in its concept and coding implementation compared to other heuristic optimization techniques.

6. CONCLUSION

In this project, a swarm optimization technique called "Firefly Optimization" is performed using the concept of adaptive Gamma Correction Method. The gamma value gives the information about the contrast of an image. Optimization is used to obtain an optimized gamma value of simple and complex images with less number of iterations in less time. With the optimized gamma value, the proposed algorithm is tested for various types of data sets related to low, high, medium resolution of pixels. From the analysis, we conclude that the proposed algorithm is better in terms of image clarity, information content and number of iterations taken by the algorithm to produce the output.

ACKNOWLEDGEMENTS

We are very glad to express our deep sense of gratitude to **Mr.Ch. Rambabu**, Senior Grade Assistant Professor of Electronics and Communication Engineering and **Dr. K. Kalyan Raj** Associate Professor of Electrical and Electronics Engineering for their guidance and cooperation in completing this project. We convey our heartfelt thanks to them for his inspiring assistance till the end of our project.

We convey our sincere and indebted thanks to our beloved Head of the Department **Dr. V. V. K. D. V. Prasad**, for his encouragement and help for completing our project successfully.

We also extend our gratitude to our Principal **Dr. P. Ravindra Babu**, for the support and for providing facilities required for the completion of our project.

We impart our heartfelt gratitude to all the Lab Technicians for helping us in all aspects related to our project.

We thank our friends and all others who rendered their help directly and indirectly to complete our project.

REFERENCES

1. Hassanpour H, Amiri SA (2011) Image quality enhancement using pixel-wise gamma correction via SVM classifier. IJE Trans B: Applications 24(4):301
2. Huang S-C, Cheng F-C, Chiu Y-S (2013) Efficient contrast enhancement using adaptive gamma correction with weighting distribution. IEEE Tranaction on Image Processing 22(3):1032–1041
3. Chiu YS, Cheng FC, Huang SC (2011) Efficient contrast enhancement using adaptive gamma correction and cumulative intensity distribution. In: Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, Anchorage, USA, pp 2946–2950
4. Amiri SA, Hassanpour H (2012) A preprocessing approach for image analysis using gamma correction. Int J Comput Appl 38(12):38–46
5. Al-Ameen Z, Sulong G, Rehman A, Al-Dhelaan A, Saba T, Al-Rodhaan M (2015) An innovative technique for contrast enhancement of computed tomography images using normalized gamma-corrected contrast- limited adaptive histogram equalization. EURASIP Journal on Advances in Signal Processing 32:1–12.