Image Processing using Ant Colony Optimisation

By:
Akhoury Pratyush Kumar - 17MT30002
Abhisek Pani - 17MT30001

Contents:

Introduction:

Images always tend to fascinate us ranging from images of microbes to images of distant galaxies, but to capture and process these images we often face many difficulties, one of which is **optimum contrast**. A human eye can often see the detailed clouds in the blue sky on a bright day yet observe the minute details in the darkest corner of the room at the same time, but try to point your phone camera on a bright day at a window a keep a flower vase or anything with enough details in front of you. Now touch on the vase in your smartphone to focus on it and click a picture, then touch on the sky outside and then click a picture, you will be surprised to notice how much details have been lost. Phone cameras usually click high contrast pictures and in situations like these it backfires. In the first photo, where you can clearly see the flower vase, try to look at the sky, it will be totally white, or as called in the image processing world, **'blown out'**, those pixels which were representing the sky, have all similar values (R=255, G=255, B=255; in easier words **white**). While the second photo you will see the sky is perfectly blue with many details of the clouds but we have totally lost the details of the flower vase which looks totally black. Thus to tackle such problems, people came up with different techniques of capturing the image which preserves the details in the brighter and the darker regions at the same time. But such images look very flat and have no contrast at all, thus requiring some basic editing to create optimum contrast. We propose **Ant colony optimization** to create contrast in which we will also include some methods from genetic algorithm to induce some elitism in out method and thus to converge faster to the solution, we will also use simulated annealing which searches locally, but at high temperature it can select worse solution to rule out local optimums. We use intensity transformation of gray scale images, to create contrast. A grayscale image is in image with no hue and saturation value but only intensity value ranging from 0 to 255. We will use a transfer function, which maps the input intensities to the output intensities. A **gamma curve** editor or just **curve** editor is often used to create contrast in many image processing software, it is a curve where we plot input intensities from 0 to 255 vs output intensity also ranging from 0 to 255. By default this curve is a straight line of y=x, which we then edit to create contrast.This simply means we change the output intensities of the image where it is needed. For creating contrast we make the darker areas more dark and the brighter regions more bright thus often creating an inverted **S-curve**.

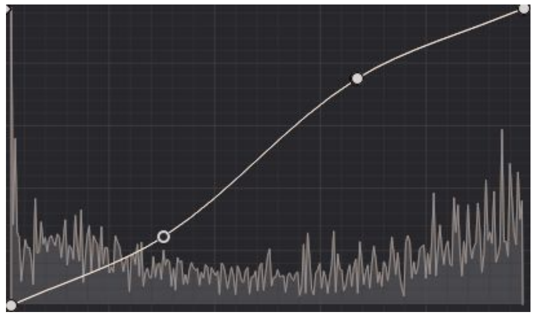

Fig: A typical S-curve where lower input intensity (dark regions) are made darker and brighter regions brighter to create contrast

Our motivation for this project was our material characterization lab experiments where we used dark field and bright field imaging along with other imaging techniques to create optimum contrast to study microstructures of materials and our interest in photography and post processing. We would be comparing our results with the general contrast slider present in an image processing software such as adobe lightroom.

PROPOSED MECHANISM:

We change the low contrast images to high contrast ones, by mapping the input intensities to the new output intensities according to a transfer function. The range of intensities in a gray scale image is between 0 and 255. Hence the transfer function generated by Ant Colony Optimization map the input intensities range from 0 to 255.
The basic of the algorithm will be as follows:

Genetic Algorithm sets the parameters of the ants by using 10 genes, then each gene is assigned to two ants making a total of 20 ants (we tried a higher number of ants but that resulted in formation of local optimas). Ant Colony Optimisation then creates 20 transfer functions, of which we select a predetermined number of output functions, and from these functions another pre-defined number of random points in each transfer function is selected. Then for each selected point simulated annealing is run another predetermined number of times, if we reach the last run of the algorithm, our output is obtained, if not we run the same process from the start again.

The pseudo code is as follow:

1. Set all reserving variables to 0.
2. Generate initial population of GA.
3. Set an initial temperature.

4. While (current iteration<iteration_max)
5. { ACO (ants generate transfer function)
6. Estimate the fitness of the transfer functions.
7. If (any of new fitness>best fitness) then take this new fitness as the best found and save its pheromone trail and transfer function.
8. If it is SA's turn, then
    - For (some of ACO's last run transfer functions)
      For (some points of selected transfer functions)
      For (a predefined number of cycles)
    a. Edit selected transfer functions
    b. Fitness calculation of new transfer function
    c. Based on P(old fitness, new fitness, temperature) decide to replace the previous transfer function and related pheromone trail and fitness with new ones.
    - If (any of new fitness>best fitness) then take this new fitness as the best found and save its pheromone trail and transfer function.
    - Decrease temperature.
    -
9. If it is GA's turn, then
    - Evaluate fitness of each individual
    - If (fitness of each individual > GA best fitness) then take new GA fitness as best found and save its related chromosomes
    - Selection followed by Reproduction and then Replacement.
    - If it is last turn of GA to assign best chromosome to parameters of all ants
10. Pheromone update }
11. Return best found transfer function as output.

Ant Colony Optimization:

The basic ant's decision for moving depends on a dynamic evaluation value (pheromone). ACO iterations depend on solution construction and pheromone update.
Generation of mapping functions is performed by ants' movement from the initial point (0,0) to the final point (255,255) in the input intensity vs the output intensity graph. Now suppose the input intensity of a given image doesn't start at 0 itself (refer to the result section of this paper to see such examples) Then any intensity<minimum-intensity is mapped to 0, i.e. output intensity for such cases is set to 0 because that is the starting

point for every ant. Similarly for such cases where the max-intensity of the input does not reach 255, then all intensities>maximum-intensities are mapped to 255. For selecting the next point to move, the roulette wheel technique is used. The selection probability (P) is presented as:

$$P = \frac{(1+\tau_i)^{\alpha} \times ((1+(\frac{k_i}{\gamma})^{10}) \times \eta_i)^{\beta}}{\sum_{i \in G(i)} (1+\tau_i)^{\alpha} \times ((1+(\frac{k_i}{\gamma})^{10}) \times \eta_i)^{\beta}}$$

where the parameters α and β control the relative importance of the pheromone trail. G(i) is the set of neighborhood points around ant. τ_i is the pheromone amount that exists in a neighbor point. In order to avoid the zero probability in areas with no pheromone, the value 1 is added to τ_i . η_i is a heuristic value that for the neighbors of an ant is set according to the following image.

0	C_{up}	1
0	ant	C_{right}
0	0	0

For up and right directions, ki is set to Intensity(input)–Intensity(input)_min (horizontal axis) and Intensity(output) (vertical axis), since we don't want our ants to move backwards but only upwards and to the right, other neighbouring positions are set to 0. The parameter γ helps ants move to target, considering if

$$\frac{k_i}{\gamma} < 1 \rightarrow \left(\frac{k_i}{\gamma}\right)^{10} < \frac{k_i}{\gamma},\ if\ \frac{k_i}{\gamma} > 1 \rightarrow \left(\frac{k_i}{\gamma}\right)^{10} > \frac{k_i}{\gamma}$$

Now as the ants move up it also moves right but to have the ant not drift along only in the upward direction we use the exponent which we used as 10, which gives the method enough power to ensure that in an ant moves excessively up, it's probability to move right increases sufficiently, which can be understood as :

$$(I_{out} \uparrow) \,\&\, (I_{out} < \gamma) \rightarrow P_{right} \uparrow$$

$$(I_{out} \uparrow) \,\&\, (I_{out} > \gamma) \rightarrow P_{right} \uparrow\uparrow$$

Same thing would happen if the ants started moving excessively to the right, this method thus ensures that the ants will reach their end point that is (255,255). However, if the ants choose a path that ends before our end point, that is suppose: (input_max,255) or (255, output_max), they will be guided toward the last point automatically. The parameters α, β, γ, C_up, and C_right are set by genetic algorithm. Also, note that the value of γ which comes from GA, is assumed for vertical direction (upside movement of ants) with the range of 255 and it needs to be adjusted automatically for horizontal direction (right side movement of ants) with the range of Intensity(input)_max-Intensity(input)_min.
20 ants seemed to be a good number of ants for this method. SA modifies the recent pheromone trail and transfer functions of ants after they have moved from the start point to the last point. Then the global pheromone update is performed as:

$$\tau_{ij}(t+1) = (1-\rho) \times \tau_{ij}(t) + \sum_{l=1}^{20} \Delta\tau_{ij}^{l}(t)$$

where ρ is the evaporation rate equal to 0.4. Δτijl is the amount of pheromone deposited between points i and j by the l-th ant.

.

GENETIC ALGO PART:

In this project, for providing a faster convergence of the ACO, we are using Genetic Algorithm in the hybrid algorithm- involving the fundamentals of encoding, selection, and reproduction. The ACO has five parameters, (namely, α, β, γ, Cup, and Cright), which are encoded as a real coded chromosome with five genes. In our case, we are assuming 20 ants and a total of 10 chromosomes. Here, each chromosome is responsible for parameter adjustment of two ants. At the end of each and every generation, the individuals in the population duplicate their values into the corresponding two ants.

Limits on variables: 0<α, β<5 100<γ<250 0<Cup,Cright<3

The Roulette Wheel technique is used for selection of parents. The GA fitness helps in testing all individuals and the two best ones are selected for breeding two offsprings. Thus, the next generation is formed by replacing ta parent and the weakest individual with the new offsprings. Hence giving us an elitist algorithm that lets only the fittest to survive.

The Fitness Function is defined as:

$F^{GA} = F^{ant1}_{mean} + F^{ant2}_{mean} + F^{ant1,2}_{best\ found}$

Where F^{GA} is the estimated fitness for each individual.

F^{ant1}_{mean} and F^{ant2}_{mean} are the average fitness of transfer functions created by the two corresponding ants in ACO's iterations run between the two GA generations.

$F^{ant1,2}_{best\ found}$ is the best-found fitness value in the mentioned procedure.

Crossover and Mutation Operators carry out the reproduction stage.
P(Uniform Crossover)=0.85
P(Mutation)=0.05, where P(i)= Probability of 'i'

SIMULATED ANNEALING PART:

The most important components of the SA method are neighborhood definition, probability function, initial temperature, and cooling schedule. Here, the SA optimizes the best-found and other transfer functions created by the last run of ACO. Also, ant's pheromone trail relative to its optimized transfer function is adjusted according to changes made in the transfer function. Via Fig.6 we can observe the various neighborhoods of the transfer function and the result of choosing any one. Hollow circles show the active points while the modified locations of currently selected points are shown by filled circles. If none of the neighbors of the current point are selected, the optimization of that point will be terminated.
The probability function for selecting a neighbor is defined as :

$$\begin{cases} P = e^{(Fnew\ -Fold)/(0.05*Fold*T)} & \text{if } F^{new} < F^{old} \\ P = 1 & \text{if } F^{new} > F^{old} \end{cases}$$

F^{old} is the current fitness
F^{new} is the resulting fitness
T is the temperature

$T_i = 200$
Cooling Equation followed: T(t+1)=T(t)*0.5

FITNESS CRITERION:

The fitness function helps in providing a bottleneck of the work from the point of views of computational cost and image quality. Hence to obtain automatic image enhancement, a fitness function independent from any human parameters should be set. Also, the function should not just involve the multiplication of operands since for weaker images, it would lead to a lower result, thus not being efficient.

Therefore, in order to overcome the above mentioned problems, the fitness function is set like this :

$F = \sqrt[3]{(STD \times ENTROPY \times SOBEL)}$

Where STD= Standard Deviation of intensities(contrast measurement)

ENTROPY= Measure of Randomness of gray levels

SOBEL is defined as : $mean(|sobel_{vertical}+sobel_{horizontal}|)$ used for edge detection of images

Results:

For testing this method we will be using one image that is underexposed, one that is normally exposed, and one that is overexposed, to test the method in different circumstances, and we will compare these results with that of the contrast slider present in an image processing software (in this case we have used adobe lightroom).

Input underexposed image:

Fig: Underexposed image of the tea shop in Tech-Market

Result from the ACO method:

Fig: Contrast using ACO.

Result using the contrast slider:

Fig: Contrast using the contrast slider in lightroom

The best path that the ant traced on the map of input vs output intensity:

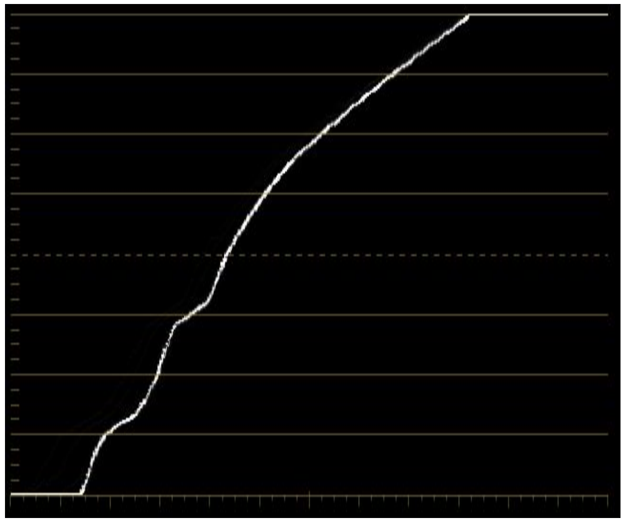

Fig: The best path (X axis is the input intensity, Y axis is the output intensity)

(We have changed the white background of the trace to black with adding grids and made the trace white for clearer visibility)

Input normally exposed image:

Fig: Normal exposure image of the main building.

Result from ACO:

Fig:Contrast with ACO method.

Result from contrast slider:

Fig: Contrast using contrast slider in lightroom

The best path that the ant traced on the map of input vs output intensity:

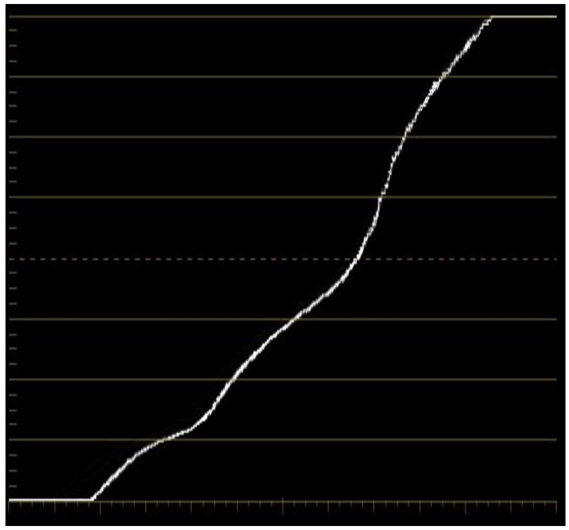

Fig: The best path (X axis is the input intensity, Y axis is the output intensity)

Input overexposed image:

Fig: overexposed image or a tree.

Result from ACO:

Fig: Contrast using ACO method

Result from contrast slider:

Fig: Contrast using contrast slider in lightroom

The best path that the ant traced on the map of input vs output intensity:

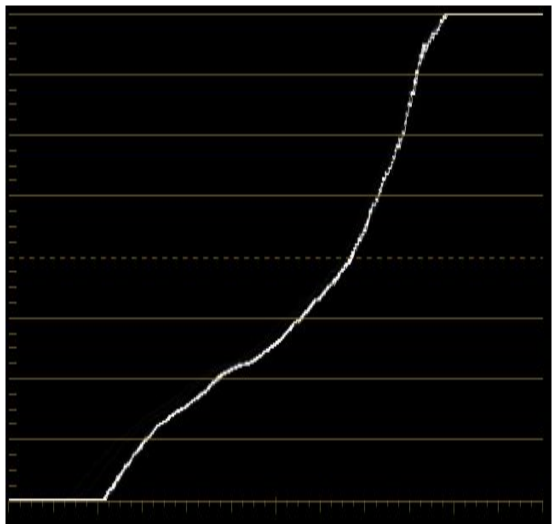

Fig: The best path (X axis is the input intensity, Y axis is the output intensity)

Conclusion:

We can see from the results that the ACO method gave better results than the general contrast slider in the underexposed image (see the sign on the left that is visible in the ACO method but barely so in the contrast slider method), also in the normally exposed image of the main building the ACO method produced a high contrast image (see how main building is in contrast with the sky). Although this does not mean that the ACO method is better than a high end image processing software. In fact, on finding out how the contrast slider works, it was justified why the ACO method produced better results.

The contrast slider basically increases the difference between the darkest and the brightest part of the image (i.e., it increases Intensity(input)_max - Intensity(input)_min) while maintaining a straight line relationship between the input intensity vs the output intensity. Whereas, in ACO method that straight line relationship between Input and output intensity is not maintained thus producing a curve that is best for every different image. Also nowadays these advanced image processing softwares allow us to edit this particular curve and see the results in real time which can help us create our customized looks, but the ACO method gives us a good start.

The simulated annealing part of the method can be disabled to provide faster results at the cost of output image quality, but for a primary contrast adjustment this method would yield desirable results.

The Final Codes can be found in our repo: https://github.com/paniabhisek26/MT21104_17MT30001_30002/tree/master

REFERENCES:

- B. Liu and P. Meng, “Hybrid algorithm combining ant colony algorithm with genetic algorithm for continuous domain,” *In Proceedings of the 9th International Conference for Young Computer Scientists, pp. 1819-1824, 2008*
- Z. J. Lee, S. F. Su, C. C. Chuang, and K. H. Liu, “Genetic algorithm with ant colony optimization (GA-ACO) for multiple sequence alignment ,” *Journal of Applied Soft Computing, Vol 8, pp. 55-78, 2008.*
- A. Rezaee, “Extracting edge of images with ant colony,” *Journal of Electrical Engineering, Vol.59, No.1, pp. 57-59, 2008.*
- R. Moussa, M. Beurton-Aimar, and P. Desbarats, “On the use of social agents for image segmentation,” *International Conference on Complex Systems and Applications(ICCSA 2009),* Le Havre, France, 2009.
- F. Saitoh, “Image contrast enhancement using genetic algorithm,” *IEEE International Conference on Systems, Man, and Cybernetics, 1999.*
- M. Braik, A. Sheta and A. Ayesh, “Image enhancement using particle swarm optimization,” *In Proceedings of the World Congress on Engineering(WCE), Vol.1, U.K, July 2007.*
- R. C. Gonzalez, and R. E. Woods, *Digital Image Processing,* Third Edition, 2008.
- R. Poli, and S. Cagnoni, “Evolution of pseudo-colouring algorithms for image enhancement with interactive genetic programming,” *Technical Report: CSRP-97-5,* Univ. of Birmingham, Jan 1997.
- T. White, B. Pagurek, and F. Oppacher, “ ASGA: improving the ant system by integration with genetic algorithm,” *In Proceedings of the Third Annual Conference Genetic Programming, 1998.*