Knowledge-based differential evolution approach to quantisation table generation for the JPEG baseline algorithm

B. Vinoth Kumar* and G.R. Karpagam

Department of Computer Science and Engineering,
PSG College of Technology,
Coimbatore – 641004, Tamil Nadu, India
Email: bvk@cse.psgtech.ac.in
Email: grkrm@cse.psgtech.ac.in
*Corresponding author

Abstract: Image quality/compression trade-off mainly depends on quantisation table used in JPEG scheme. Therefore, the generation of quantisation table is an optimisation problem. Even though recent reports reveal that classical differential evolution (CDE) is a promising algorithm to generate the optimal quantisation table, it is slow in convergence rate due to its weak local exploitation ability. This paper proposes knowledge-based differential evolution (KBDE) algorithm to search the optimal quantisation table for the target bits/pixel (bpp). KBDE incorporates the image characteristics and knowledge about image compressibility in CDE operators to accelerate the search. KBDE and CDE algorithms have been experimented on variety of images and an extensive performance analysis has been made between them, which reveal that KBDE accelerates the convergence rate of CDE without compromising on the quality of solution. Further, a statistical hypothesis test (t-test) confirms the result.

Keywords: image compression; joint photographic experts group; JPEG; quantisation table; optimisation; meta-heuristic search; differential evolution; knowledge-based differential evolution; KBDE; selection pressure; statistical hypothesis test; t-test.

Reference to this paper should be made as follows: Vinoth Kumar, B. and Karpagam, G.R. (2016) 'Knowledge-based differential evolution approach to quantisation table generation for the JPEG baseline algorithm', *Int. J. Advanced Intelligence Paradigms*, Vol. 8, No. 1, pp.20–41.

Biographical notes: B. Vinoth Kumar is a Senior Grade Assistant Professor with 11 years of experience in the Department of Computer Science and Engineering, PSG College of Technology, Coimbatore. His areas of research include soft computing and digital image processing. He has published his research contributions in peer reviewed international journals and conferences. He has served as a member of the board of studies and reviewer in conferences.

G.R. Karpagam is a Professor with 18 years of experience in the Department of Computer Science and Engineering, PSG College of Technology, Coimbatore. She is an IEEE member doing her research in the field of web services, evolutionary computing, MDA and cloud. She has published papers in peer reviewed national/international journals and conferences and a reviewer of international journals. She has received research grants from Indian

Government agencies and has setup IaaS data centre for cloud research. She has established industry funded laboratories namely open-source software, SOA and cloud, in collaboration with leading MNCs.

1 Introduction

Joint photographic experts group (JPEG) is a highly dominating still image compression standard in the cyberspace since 1992. A significant amount of images on the internet is in JPEG format (Vinoth Kumar and Karpagam, 2015). In algorithmic point of view, JPEG supports four operation modes; sequential discrete cosine transform (DCT), progressive DCT, lossless and hierarchical. Among these modes, sequential DCT is referred as baseline JPEG algorithm where an image is subdivided into 8 × 8 blocks and forward DCT is applied to each block to obtain 8 × 8 DCT coefficients. These DCT coefficients are quantised by 8 × 8 quantisation table which is recommended by JPEG group. Then, entropy encoding is applied on quantised DCT coefficients and the result is stored as a compressed file. An exact reverse operation of the above said process leads to JPEG decompression. Wallace (1992) provides a detailed report of the JPEG baseline algorithm.

Quantisation table used in the JPEG standard is very crucial in image quality/compression trade-off and also JPEG standard allow the users to customise it according to their applications. Therefore, generation of quantisation table is an open optimisation problem. Many researchers use meta-heuristic techniques such as simulated annealing (Sherlock et al., 1994), genetic algorithm (Wu, 2004; Lazzerini et al., 2010; Vinoth Kumar and Karpagam, 2014), particle swarm optimisation (Ma and Zhang, 2012), firefly algorithm (Milan and Nebojsa, 2014) and differential evolution (DE) (Vinoth Kumar and Karpagam, 2015) to optimise the quantisation table. Classical differential evolution (CDE) is proved as a newly promising optimisation technique in the design of quantisation table for the JPEG baseline algorithm (Vinoth Kumar and Karpagam, 2015). Even though CDE maintains the global search capability, it suffers from weak local exploitation ability which in turn decelerates the convergence rate (Babu and Rakesh, 2006; Mashwani, 2014). Many researchers have made their attempts to modify the scaling factor (Ching-Hung, 2012), mutation rule (Xuemei, 2010; Kumar et al., 2011; Ali et al., 2013), crossover strategies (Zhenyu et al., 2008; Sandeep et al., 2014) to accelerate the convergence rate. Vinoth Kumar and Karpagam (2014) have proved that the incorporation of application specific domain knowledge in genetic algorithm accelerates the convergence rate. Similarly, the incorporation of domain knowledge in DE algorithm was done for different applications (Chen and Yang, 2013, 2014) to the best of our knowledge; it has never been used for this application. Therefore, in this paper, a knowledge-based differential evolution (KBDE) algorithm is proposed to optimise the quantisation table for a JPEG baseline algorithm.

This paper incorporates the knowledge proposed by Vinoth Kumar and Karpagam (2014) into CDE algorithm and our contributions lie in the ways to handle the knowledge in CDE algorithm.

1 knowledge is used to initialise the chromosome in which gene values are limited in the range of 1 to 255

2 incorporation of knowledge to introduce the variable pressure in parent selection at differential mutation step

3 knowledge is incorporated in crossover operation to introduce the knowledge-based crossover (KBC) rate.

Our objective is to accelerate the convergence rate of CDE without any loss in the performance. To this objective, the image characteristics and knowledge about image compressibility is incorporated in CDE operators, namely knowledge-based initialisation (KBI), knowledge-based mutation (KBM) and KBC to significantly enhance the convergence rate. In order to compare the results with CDE, KBDE has been experimented with same images taken in Vinoth Kumar and Karpagam (2015). Also the performance comparison of both CDE and KBDE has been performed in terms of average best unfitness value, average best of generations, optimisation accuracy, likelihood of evolution leap, probability of convergence, average number of function evaluations (AFES) and successful performance (SP). The results prove that KBDE achieves a faster convergence rate than CDE without any loss in the quality of the solution and the same is confirmed by using a statistical hypothesis test.

The rest of this paper is organised as follows. A brief review of the CDE algorithm is given in Section 2. Section 3 illustrates the proposed KBDE. The various performance measures are explained in Section 4. The experiments and results are discussed in Section 5. Final thoughts are concluded in Section 6.

2 Classical differential evolution

The DE is a stochastic, population-based optimisation algorithm introduced by Storn and Price in 1995. DE is successfully applied in many fields such as engineering, statistics and finance. Initialisation, mutation, crossover and selection are the four important steps in DE. By changing these operators, there are many variants of DE in which DE/Rand/1/bin is termed as classical DE (Vinoth Kumar and Karpagam, 2015). An initial population of N_p chromosomes is generated randomly. During mutation, three random chromosomes $x_{r1,G}$, $x_{r2,G}$, and $x_{r3,G}$ in the current population G are taken to calculate the mutant chromosome $v_{i,G}$ as shown in equation (1). F is a scaling factor which is taken between 0 and 1 to control the evolution. The mutant chromosome $v_{i,G}$ performs binomial uniform crossover with target chromosome $x_{i,G}$ to form trial chromosomes as shown in equation (2). Crossover is done based on crossover probability C_r which takes between 0 and 1 that decides the number of genes of mutant chromosome contribute to the trial chromosome $u_{i,G}$. Now the fitness value is calculated for trial chromosome and if it is less than fitness value of target chromosome (in case of minimum fitness function), then the target chromosome is replaced by it for the next generation $G + 1$, otherwise, target chromosome for the next generation is retained as shown in equation (3). The pseudo code of CDE is given in Algorithm 1.

$$v_{i,G} = x_{r1,G} + F\left(x_{r2,G} - x_{r3,G}\right) \quad (1)$$

$$u_{i,G} = u_{j,i,G} = \begin{cases} v_{j,i,G} & \text{if } \left(rand_j(0,1) \le C_r \text{ or } j = j_{rand} \right) \\ x_{j,i,G} & \text{otherwise} \end{cases} \tag{2}$$

$$x_{i,G+1} = \begin{cases} x_{i,G} & \text{if } \left(fitness\left(x_{i,G}\right) \le fitness\left(u_{i,G}\right) \right) \\ u_{i,G} & \text{otherwise} \end{cases} \tag{3}$$

Algorithm 1. Classical differential evolution-pseudo code adopted from Vinoth Kumar and Karpagam (2015)

Initialize population of chromosomes randomly;
Evaluate the chromosomes;
While Maximum Generation not reached *do*
  For all chromosomes *do*
    Select the target chromosome;
    Choose three chromosomes in the population randomly;
    Compute the mutant chromosome;
    Perform crossover between the target and mutant chromosomes to form trial chromosome;
    Evaluate the trial chromosome;
    Replace target chromosome by trial chromosome if unfitness value of trial chromosome is smaller than target chromosome;
  End for
End while
Return best chromosome;

3 Knowledge-based differential evolution (KBDE)

Although the CDE algorithm is good in global search capability, it has a low convergence rate. The image characteristics and knowledge about image compressibility can be used in CDE to accelerate the convergence rate.

3.1 Knowledge about image compressibility

The information about the JPEG baseline algorithm and quantisation table can be used as a knowledge-base which is incorporated in operators to accelerate the search (Vinoth Kumar and Karpagam, 2014). The JPEG baseline algorithm applies DCT on 8 × 8 blocks to form 8 × 8 DCT coefficients, in which the energy of the signal concentrates in the top left of the block. By having a very few top left (low frequency) DCT coefficients, the whole block can be reconstructed without much loss. On examining the quantisation table given in Annex K of JPEG standard, it is observed that the values in the top left are less when compared to other values in the quantisation table. This shows that the quantisation process in the JPEG process tries to retain the DCT coefficients in the top left of the block with minimum loss.

From the above discussion, the following are observed as knowledge (Vinoth Kumar and Karpagam, 2014).

1 the low frequency DCT coefficients are very important to maintain the quality of the block

2 the left top of a quantisation table should have less value and also it play an important role than right bottom values

3 DC and AC coefficients give the characteristics of an image block.

The above said knowledge is incorporated in operators such as initialisation, mutation and crossover.

3.2 Knowledge-based initialisation (KBI)

KBI is adopted from Vinoth Kumar and Karpagam (2014) where each quantisation table, denoted as a chromosome, is divided into four 4 × 4 sub-tables as shown in Figure 1. As discussed in the previous sub-section, the values in top left sub-table should be less than the right bottom sub-table. The range of values for the top left, top right and bottom left sub-tables are set between 1 to 145 and for a bottom right sub-table is set between 40 to 255. Chromosomes are generated randomly based on their corresponding sub-table range of values. Hence, knowledge base is used to create a better initial population rather than random ones.

Figure 1 Sub-table view in 8 × 8 quantisation table

4 × 4 left top sub-table	4 × 4 right top sub-table
4 × 4 left-bottom sub-table	4 × 4 right bottom sub-table

3.3 Evaluation

The objective is to generate the quantisation table that produces better decoded quality for the target bits/pixel. The same unfitness function which was used in Vinoth Kumar and Karpagam (2015) is taken here to find the survival probability of each chromosome. An unfitness function is given in equation (4).

$$\xi = a\left(\frac{8}{B_r} - \lambda\right)^2 + \varepsilon \tag{4}$$

where $a = 10$, Br = bits/pixel, ε = mean squared error, λ = desired compression ratio $= \left(\frac{8}{\text{target bits per pixel}}\right)$.

3.4 Knowledge-based mutation (KBM)

Generally to increase the convergence rate, the strategies like DE/current-to-best/*n* and DE/best/*n* can be used to guide the search process faster towards the best solution. But the drawback is the diversity loss in the population (Das et al., 2009; Ankush et al., 2011). To overcome this situation, many researchers have developed a significant number of new mutation strategies (Asafuddoula et al., 2011; Wenyin et al., 2011). However, there is an equal chance for all chromosomes to be selected as parents to form the mutant chromosome. Many researchers have made their attempts to increase the selection pressure (Andrew et al., 2007; Islam et al., 2012). However, to the best of our knowledge, the incorporation of image characteristics to increase the selection pressure is never attempted in differential mutation. In this paper, a deterministic approach is used to select the parents from the top ranked chromosomes. This approach introduces the variable selection pressure in differential mutation step.

This approach maintains the top N ranked chromosomes from the population and in that, the chromosomes, which produce better decoded quality, are selected as parents. Computation of the decoded quality of each chromosome for an image increases the computational time drastically. To overcome this, the decoded quality is calculated only for the representative image blocks of an image. K means algorithm with deterministic centroid initialisation method (Vinoth Kumar and Karpagam, 2014; Vinoth Kumar et al., 2015) is used to cluster the 8 × 8 image blocks of an image and the block which is close to the centroid is taken as a representative block of that cluster. The blocks are clustered based on their DC and standard deviation of AC coefficients. Here knowledge about the image is used to introduce the variable pressure in parent selection, which accelerates the search towards the best solution.

3.5 Knowledge-based crossover (KBC)

The crossover operator combines the genes of mutated and target chromosome, according to the crossover rate *Cr* which is also called as mutation probability (Daniela, 2009). According to the knowledge-base, if the amount of change in the left top sub-table of trial chromosome is high, then the chance of being different from the target chromosome will be high which may lead to the better result than target chromosome. Generally C_r is fixed to all gene positions in a chromosome. In order to introduce more randomness in a left top sub-table, a KBC rate is proposed where it varies based on their gene position *p* of each chromosome. It is calculated as shown in equation (5) and the result is truncated to one decimal point. Gene positions of a chromosome (Vinoth Kumar and Karpagam, 2014) are shown in Figure 2. KBC rate is high for left top of the chromosome; due to this, there is a more chance of crossover between mutated and target chromosome. In another perspective, the KBC rate decreases the crossover rate from left top to right bottom of a chromosome. Hence, there is a more chance of crossover only in left top which can be viewed as selective mutation. Here knowledge is contributed to guide the search in the feasible region. The pseudo code for KBDE is given in Algorithm 2.

$$C_r = truncate\left(1 - \frac{p}{64}\right) \tag{5}$$

Figure 2 8 × 8 chromosome – gene positions

1	3	4	10	11	21	22	36
2	5	9	12	20	23	35	37
6	8	13	19	24	34	38	49
7	14	18	25	33	39	48	50
15	17	26	32	40	47	51	58
16	27	31	41	46	52	57	59
28	30	42	45	53	56	60	63
29	43	44	54	55	61	62	64

Algorithm 2. Knowledge-based differential evolution-pseudo code

```
Generate population of chromosomes using KBI;
Evaluate the chromosomes;
While Maximum Generation not reached do
    Take superior N chromosomes based on their low unfitness value
    Select the chromosomes which produce better image decoded quality and form
    sub-population
    For all chromosomes do
        Select the target chromosome;
        Choose three chromosomes in the sub-population randomly;
        Compute the mutant chromosome;
        Perform KBC between the target and mutant chromosomes to form trial chromosome;
        Evaluate the trial chromosome;
        Replace target chromosome by trial chromosome if unfitness value of trial chromosome
        is smaller than target chromosome;
    End for
End while
Return best chromosome;
```

The knowledge injected to CDE operators like initialisation and crossover are common to all images and the knowledge injected in differential mutation step is specific to an input image which is extracted automatically and applied into it.

4 Performance measures

In order to compare the performance of KBDE with CDE, the following measures are considered in this study. These measures are taken from Vinoth Kumar and Karpagam (2015). The first two measures are used to validate the quality of the decompressed image and the remaining measures are used to validate the efficiency of CDE and KBDE algorithm.

4.1 Mean squared error MSE

This is the most commonly used measure which finds the average of squared error between the original and decompressed images. It is given by

$$MSE(X,Y) = \frac{1}{M.N} \sum_{i=1}^{M} \sum_{j=1}^{N} \left(X_{ij} - Y_{ij} \right)^2 \tag{6}$$

4.2 Peak signal to noise ration PSNR

This measure tells the ratio between the maximum possible power of a signal and the power of error signal. It tells how close the reconstructed image to the original image. It is defined as

$$PSNR = 10 \log_{10} \left(\frac{MAX_O^2}{MSE} \right) \tag{7}$$

4.3 Average best unfitness value $f_a(k)$

This measure calculates the best unfitness value after k generations, averaged over n independent runs. It is given by

$$f_a(k) = \frac{\sum_{runs=1}^{n} Best\ Unfitness\ value(k)}{n} \tag{8}$$

4.4 Average best-of-generation $\overline{BOG}$

This measure considers the best unfitness value over all k generation in total number of runs n. It is calculated as shown in equation (9). This measure is used to analyse the entire optimisation process of an evolutionary algorithm

$$\overline{BOG} = \frac{1}{n} \frac{1}{k} \sum_{r=1}^{n} \sum_{g=1}^{k} f\left(BOG_{rg} \right) \tag{9}$$

where $f(BOG_{rg})$ expresses the unfitness value of the best solution at generation g of run r (among n independent runs).

4.5 Optimisation accuracy Acc_k

This measure finds the location of the best found solution between the worst known solution in search space Min_s and best known solution in search space Max_s as shown in equation (10). This measure value may vary from 0 (worst) to 1 (best). This measure used to analyse the accuracy of an evolutionary algorithm.

$$Acc_k = \frac{f_a(k) - Min_s}{Max_s - Min_s} \tag{10}$$

4.6 Likelihood of evolution leap Lel(k)

Evolutionary leap is observed as an improvement in solution in two successive generations. This measure calculates the number of leaps l in k generations averaged over n independent runs as shown in equation (11). This measure is used to analyse the search capability of an evolutionary algorithm.

$$Lel(k) = \frac{l}{n} \tag{11}$$

4.7 Probability of convergence P

This measure tells that how many independent runs are able to achieve the optimal/ sub-optimal solution. It varies from 0 to 1 and higher value is preferred. This measure is used to analyse the consistency of an evolutionary algorithm. It is given by the equation (12).

$$P = \frac{s}{n} \tag{12}$$

where

s number of successful independent runs

n total number of independent runs.

4.8 Average number of function evaluations AFES

This measure calculates the number of function evaluations to reach the optimal/ sub-optimal solution in each successful run as shown in equation (13). This measure is used to analyse the convergence speed of an evolutionary algorithm. The lower value is preferred.

$$AFES = \frac{1}{s}\sum_{i=1}^{s} EVAL_i \tag{13}$$

where $EVAL_i$ = number of function evaluation in the successful run i.

4.9 Successful performance SP

This measure calculates the ratio between the AFES and probability of convergence. The lower value is preferred for this measure and it is shown in equation (14).

$$SP = \frac{AFES}{P} \tag{14}$$

5 Experimental results and discussion

Although CDE is a promising algorithm for the generation of quantisation table, it is necessary to accelerate the convergence rate. The objective of this paper is to accelerate the convergence rate of CDE without any loss in the quality of the solution. Here image characteristics and knowledge about image compressibility are incorporated in the CDE operators to achieve the objective. The programs are realised using MATLAB R2008b and they are implemented on a Dell workstation of Intel® Xenon® CPU E3-1240 V3 @ 3.40 GHz processor with 16 GB of RAM. Benchmark images shown in Figure 3 are taken from USC-SIPI Image database which is of size 256 × 256 and digitised to 256 gray levels.

Figure 3 Uncompressed test images, (a) Lena (b) camera man (c) Barbara (d) couple (e) crowd (f) bridge (g) clock (h) baboon (i) pattern (j) montage

(a) (b) (c) (d)

(e) (f) (g) (h)

(i) (j)

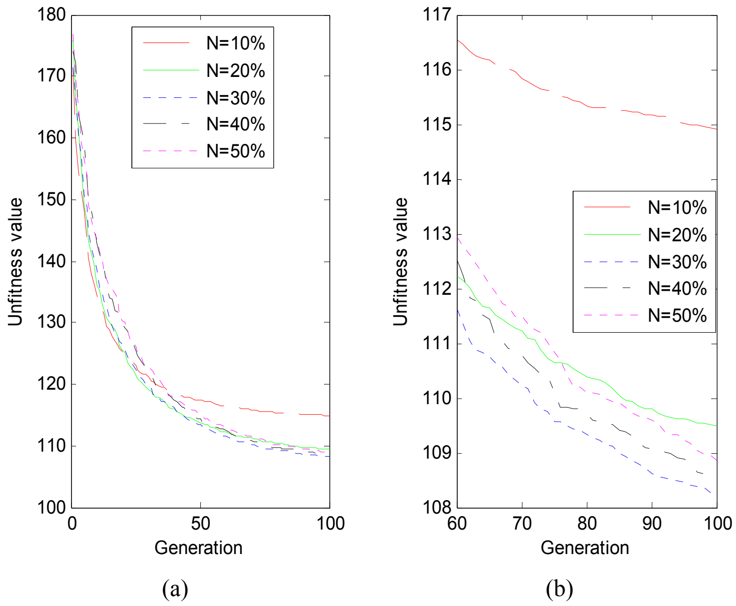
Image block clustering process plays a vital role in KBM to select the chromosomes as parents. Image is partitioned into 8 x 8 blocks and each block is identified by its DC coefficient value and Standard deviation AC coefficient values. All blocks are grouped into K clusters using k means algorithm and the block which is close to its cluster centroid is taken as representative block of each cluster. In KBM, top N chromosomes are selected based on their unfitness value and among them; the chromosomes, which produce a better decoded quality for the representation blocks are selected as parents. Therefore the selection of parents depends on total number of clusters K and top N chromosomes. An empirical analysis is performed on different number of clusters in the image block clustering process; Vinoth Kumar and Karpagam (2014) and Vinoth Kumar et al. (2015) proved that the total number of clusters at 100 produces the best result. Therefore, in our experiment, the total number of clusters is set at 100. The selection of top N chromosomes is crucial in the parent selection due to the following reasons:

1 if the value of N is low, then there may be low selection diversity which may lead to premature convergence

2 if the value of N is high, then it resemble the ordinary mutation which may lead to poor convergence rate.

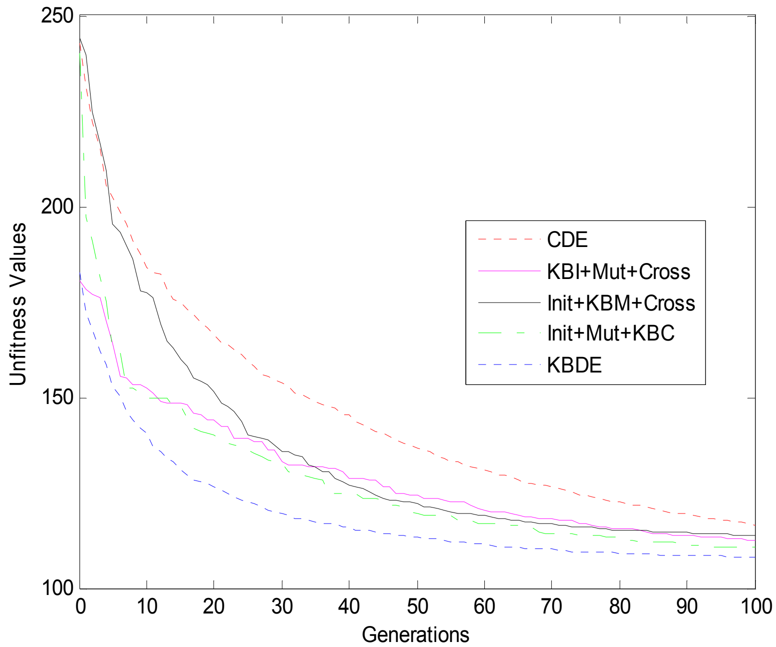
Hence there is a need to find the suitable N value. In this study, N value is chosen as 10%, 20%, 30%, 40% and 50% and N value above 50% resembles the ordinary mutation. In order to find the suitable N value, KBDE with different N values are experimented on three different complexity images Lena, Montage and Baboon with target bits per pixel set at 1.0. Figures 4(a) and 4(b) shows the unfitness value progression of the best chromosomes for different N values. Figure 4(b) is a zoomed part of Figure 4(a) at the later generations. Each point indicates the average of 20 runs. It shows top 30% gives the better result than all other N values. Therefore, in our experiment, top 30% chromosomes are used to select the parents for the mutation.

Figure 4 Unfitness value progression for different n values (see online version for colours)



To show the efficacy of knowledge-base, the following three methods are considered here in which the knowledge is incorporated in any one of CDE operators. The method 1 has knowledge only in initialisation part and it is denoted as 'KBI + Mut + Cross'; method 2 has knowledge only in mutation part and it is denoted as 'Init + KBM + Cross'; method 3 has knowledge only in crossover part and it is denoted as 'Init + Mut + KBC'. The above said methods experiment with same three images for the target bits per pixel set at 1.0. The unfitness value progression of all methods along with CDE and KBDE is shown in Figure 5. From Figure 5, it is clearly shown that knowledge in any one CDE operator accelerates the search speed. Also, it shows the knowledge incorporated in all CDE operators (i.e., KBDE) accelerates the search speed drastically. In addition KBDE achieve the optimal solution in very less generations than CDE, which provide evidence of domain knowledge.

Figure 5 Unfitness value progression to show the efficacy of knowledge-base (see online version for colours)



To evaluate the performance of the KBDE, it is experimenting with different images shown in Figure 3. The simulation parameters of CDE (Vinoth Kumar and Karpagam, 2015) and KBDE are shown in Table 1 and unfitness function for both the algorithms is same. The optimal quantisation table generated by KBDE is compared with default JPEG and CDE-based quantisation table in terms of mean squared error (MSE) and peak signal to noise ratio (PSNR). Figure 6 shows the number of chromosomes chosen as parents for KBM in each generation for KBDE. It shows the variable selection pressure given by the deterministic approach in KBM.

Table 1 Simulation parameters of CDE and KBDE

Parameter	*CDE*	*KBDE*
Population size	64	64
Scaling factor	0.3	0.3
Crossover probability	0.8	Decreasing from 0.9
Generations	150	100
Number of independent runs	20	20

Figure 6 Lena: parent selection vs. generation (target bits/pixel = 1.0)

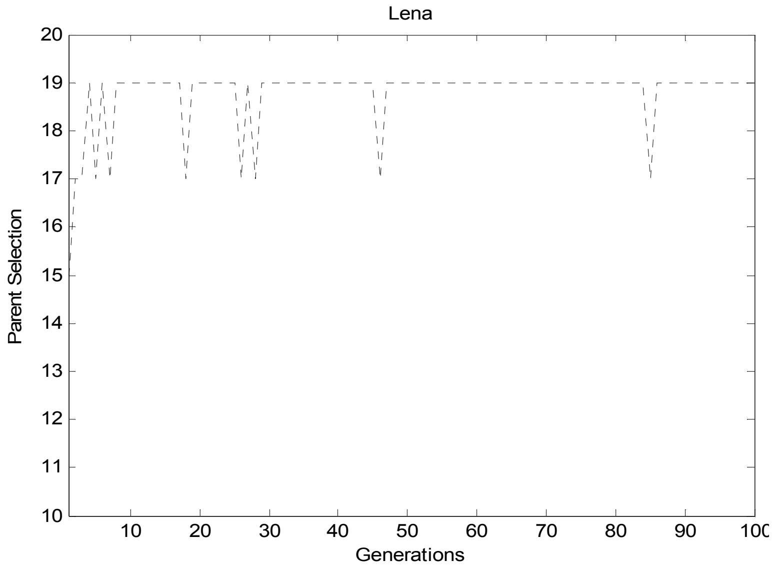


Table 2 displays the mean results of both CDE and KBDE for different bits per pixel 0.75, 1.0 and 1.5. From Table 2, it is clearly shown that KBDE-based quantisation table yields similar results as a CDE in terms of MSE and PSNR. However, for the images like a clock and montage in 1.5 bpp, both the algorithms need few more generations to achieve the optimal solutions in each run. Due to the incorporation of knowledge, KBDE requires an additional computational cost; however KBDE achieves the feasible solution in very less generation than CDE. In our experiments, CDE and KBDE takes 9,328.36 seconds and 6,619.91 seconds on average to achieve the feasible solution, respectively, which shows KBDE has reduced computational time of 29% than CGA.

To analyse the performance of CDE and KBDE in detail, the measures given in Section 5 have been taken into consideration. These measures are calculated for ten different images shown in Figure 1 with three different target bits/pixel 0.75, 1.0 and 1.5 in 20 independent runs. The mean results of the performance measures are given from Tables 3 to 13.

Table 3 shows the average best unfitness value after the 50th and 100th generations for both CDE and KBDE and it is shown in target bits/pixel set at 1.0. From Table 3, it is clearly seen that the unfitness values of KBDE for each image are lower than CDE. The average value of this measure for different bits/pixel is given in Table 4 which shows the average unfitness value of KBDE after 50 generations is less than the average unfitness value of CDE after 100 generations. It confirms that KBDE is converging faster than CDE.

Table 2 Comparison of image quality measures for different target bits/pixel

Target bits/pixel		*0.75*			*1*			*1.5*		
Image	*Quantisation table*	*Bits/pixel*	*MSE*	*PSNR in dB*	*Bits/pixel*	*MSE*	*PSNR in dB*	*Bits/pixel*	*MSE*	*PSNR in dB*
Lena	JPEG	0.76	51.96	31.01	1.03	34.29	32.81	1.50	19.26	35.31
	CDE	0.77	42.51	31.88	1.00	28.95	33.55	1.52	15.16	36.36
	KBDE	0.77	40.13	32.13	1.01	26.04	34.01	1.52	15.04	36.39
Camera man	JPEG	0.75	66.24	29.95	1.02	44.25	31.71	1.51	22.29	34.68
	CDE	0.77	50.87	31.10	1.01	32.41	33.05	1.52	14.26	36.62
	KBDE	0.78	47.62	31.39	1.02	28.98	33.54	1.53	13.87	36.74
Barbara	JPEG	0.77	61.71	30.26	1.01	41.93	31.94	1.50	16.92	35.88
	CDE	0.76	47.54	31.39	1.02	27.07	33.84	1.53	11.96	37.38
	KBDE	0.77	46.37	31.50	1.02	26.08	34.00	1.54	12.46	37.21
Clock	JPEG	0.75	24.28	34.31	1.00	14.64	36.51	1.51	7.21	39.58
	CDE	0.74	21.04	34.93	0.99	14.81	36.46	1.49	7.76	39.27
	KBDE	0.75	17.77	35.67	0.99	11.65	37.51	1.50	7.57	39.37
Bridge	JPEG	0.75	157.67	26.19	1.03	120.02	27.37	1.61	75.77	29.37
	CDE	0.77	142.36	26.63	1.05	99.33	28.19	1.55	60.92	30.31
	KBDE	0.79	142.42	26.63	1.11	98.20	28.24	1.58	57.62	30.56

Table 2 Comparison of image quality measures for different target bits/pixel (continued)

Target bits/pixel		*0.75*			*1*			*1.5*		
Image	*Quantisation table*	*Bits/pixel*	*MSE*	*PSNR in dB*	*Bits/pixel*	*MSE*	*PSNR in dB*	*Bits/pixel*	*MSE*	*PSNR in dB*
Couple	JPEG	0.75	49.57	31.21	1.01	34.31	32.81	1.50	19.49	35.27
	CDE	0.75	44.60	31.67	1.00	29.70	33.43	1.52	15.60	36.23
	KBDE	0.76	43.46	31.78	1.01	28.46	33.62	1.55	14.56	36.53
Crowd	JPEG	0.76	40.51	32.09	1.01	27.25	33.81	1.52	14.77	36.47
	CDE	0.75	38.34	32.33	1.02	23.71	34.41	1.52	12.38	37.24
	KBDE	0.76	37.44	32.43	1.03	22.53	34.64	1.55	12.50	37.20
Baboon	JPEG	0.75	404.18	22.10	1.04	330	22.98	1.54	223.28	24.68
	CDE	0.78	350.59	22.72	1.06	244.48	24.28	1.51	132.89	26.94
	KBDE	0.79	349.09	22.74	1.09	234.69	24.46	1.54	129.37	27.05
Pattern	JPEG	0.75	58.63	30.48	1.01	48.10	31.34	1.51	35.91	32.61
	CDE	0.74	52.20	30.98	1.01	40.08	32.13	1.55	25.57	34.09
	KBDE	0.75	50.16	31.16	1.00	39.64	32.18	1.58	23.36	34.48
Montage	JPEG	0.75	25.20	34.15	1.00	13.54	36.85	1.50	5.51	40.76
	CDE	0.75	19.85	35.19	1.00	12.15	37.32	1.50	7.63	39.40
	KBDE	0.76	15.90	36.15	1.02	10.29	38.05	1.52	7.40	39.49

Table 3 Average unfitness values for bits/pixel = 1.0

bpp = 1.0	*CDE*		*KBDE*	
Images	*After 50 generations*	*After 100 generations*	*After 50 generations*	*After 100 generations*
Lena	44.63	33.89	29.48	26.89
Camera man	55.63	41.88	33.45	30.20
Barbara	44.75	31.48	28.56	27.33
Clock	27.54	19.86	13.81	11.78
Bridge	124.08	111.08	107.03	104.98
Couple	46.65	34.63	30.97	29.12
Crowd	37.11	27.10	23.88	23.08
Baboon	333.91	298.50	296.83	286.70
Pattern	52.89	44.50	40.77	40.07
Montage	24.76	16.56	13.45	10.64
Average	79.19	65.95	61.82	59.08

Table 4 Summary of average unfitness value for various bits/pixel

bpp	*CDE*		*KBDE*	
	After 50 generations	*After 100 generations*	*After 50 generations*	*After 100 generations*
0.75	103.51	88.88	86.42	83.56
1	79.19	65.95	61.82	59.08
1.5	55.27	40.82	40.04	35.35
Average	79.33	65.22	62.76	59.33

Table 5 Average best of generations for bits/pixel = 1.0

bpp = 1.0	*CDE*		*KBDE*	
Images	*After 50 generations*	*After 100 generations*	*After 50 generations*	*After 100 generations*
Lena	63.95	38.41	38.28	27.77
Camera man	75.06	48.66	44.55	31.28
Barbara	70.46	36.38	36.81	27.67
Clock	41.99	22.82	21.65	12.54
Bridge	157.73	115.59	120.97	105.74
Couple	63.63	39.36	39.02	29.72
Crowd	53.39	30.90	31.51	23.34
Baboon	392.21	313.08	322.38	290.22
Pattern	65.78	48.86	48.74	40.26
Montage	38.39	19.87	22.58	11.65
Average	102.26	71.39	72.65	60.02

Average best-of-generation for the periods 1st to 50th generation and 51st to 100th generation is shown in Table 5 and its average value for different bits/pixel is summarised in Table 6. From Tables 5 and 6, it is clearly seen that the average of the best unfitness value of KBDE in each period is lesser than CDE. It confirms that KBDE is better than CDE for the entire optimisation process.

Table 6 Summary of average best of generations for various bits/pixel

bpp	*CDE*		*KBDE*	
	Generations		*Generations*	
	1 to 50	*51 to 100*	*1 to 50*	*51 to 100*
0.75	126.22	94.51	96.93	84.50
1	102.26	71.39	72.65	60.02
1.5	81.51	46.42	60.16	36.93
Average	103.33	70.77	76.58	60.49

Table 7 Optimisation accuracy for bits/pixel = 1.0

bpp = 1.0	*CDE*		*KBDE*	
Images	*After 50 generations*	*After 100 generations*	*After 50 generations*	*After 100 generations*
Lena	0.82	0.92	0.97	1.00
Camera Man	0.79	0.90	0.97	1.00
Barbara	0.87	0.97	0.99	1.00
Clock	0.83	0.91	0.97	0.99
Bridge	0.84	0.96	0.98	0.99
Couple	0.82	0.94	0.98	1.00
Crowd	0.85	0.95	0.99	1.00
Baboon	0.76	0.94	0.95	0.99
Pattern	0.84	0.94	0.98	0.99
Montage	0.84	0.93	0.96	0.99
Average	0.82	0.94	0.97	1.00

Table 8 Summary of optimisation accuracy for various bits/pixel

bpp	*CDE*		*KBDE*	
	After 50 generations	*After 100 generations*	*After 50 generations*	*After 100 generations*
0.75	0.80	0.94	0.98	1.00
1	0.82	0.94	0.97	1.00
1.5	0.87	0.96	0.97	1.00
Average	0.83	0.95	0.97	1.00

Table 7 shows the optimisation accuracy value after 50th and 100th generations for target bits/pixel set at 1.0. Average value of this measure for different bits/pixel is summarised in Table 8. From Tables 7 and 8, it has been noted that the accuracy value of KBDE in

50th generation is higher than a 100th generation of CDE. It shows KBDE is very close to an optimal solution in a very less generation.

Table 9 Likelihood of evolution leap for bits/pixel = 1.0

bpp = 1.0	*CDE*		*KBDE*	
Images	*After 50 generations*	*After 100 generations*	*After 50 generations*	*After 100 generations*
Lena	1.00	1.58	1.70	3.00
Camera Man	0.90	1.60	1.63	2.90
Barbara	1.28	2.35	1.50	3.05
Clock	0.82	1.37	1.45	2.80
Bridge	1.03	1.90	1.55	2.90
Couple	0.78	1.63	1.80	3.28
Crowd	1.05	2.03	1.55	2.68
Baboon	1.13	2.38	1.40	2.53
Pattern	0.80	1.33	1.60	2.98
Montage	1.00	1.70	1.60	2.68
Average	0.98	1.78	1.58	2.88

Table 9 shows the $Lel(k)$ up to the 50th and 100th generations for target bits/pixel set at 1.0. Average values of this measure for different bits/pixel are summarised in Table 10. From Table 9 and 10, it is interesting to notice that KBDE has almost 35% more leaps than CDE in each period. Although the difference in average unfitness values of KBDE between 50th and 100th generation is less, it is able to produce more number of leaps in later generations. It shows that KBDE has the highest search capability than CDE, which helps to reach the optimal solution in less generation.

Table 10 Summary table likelihood of evolution leap for various bits/pixel

bpp	*CDE*		*KBDE*	
	After 50 generations	*After 100 generations*	*After 50 generations*	*After 100 generations*
0.75	0.75	1.56	1.44	2.62
1	0.98	1.78	1.58	2.88
1.5	1.19	2.15	1.56	2.82
Average	0.97	1.83	1.52	2.77

Table 11 shows the probability of convergence P for different bits/pixel. The P measure value of CDE shows that it does not able to reach the optimal solution at all runs for all images within the given number of generations, whereas KBDE is able to reach the optimal solution at all runs for all images except for the clock and montage images. *AFES* values and *SP* for different bits/pixel is shown in Tables 12 and 13 respectively. These measures prefer the lower values. From Tables 12 and 13, it is clear that KBDE is able to reach the optimal solution consistently within a less number of generations. *AFES* and *SP* measures of CDE could not be calculated for some images, since it is not able to reach the optimal solution in the given number of generations. For the images like a clock and montage, *AFES* value in 1.5 bits/pixel is high and it is close to the maximum number of generations. It shows that these images need fewer more generations to guarantee the feasible solution at every run.

Table 11 Probability of convergence for various bits/pixel

Algorithm	*CDE*			*KBDE*		
bpp	*0.75*	*1*	*1.5*	*0.75*	*1*	*1.5*
Lena	1	0.2	0.6	1	1	1
Camera man	1	1	1	1	1	1
Barbara	1	1	1	1	1	1
Clock	0.2	0	0	1	1	0.2
Bridge	1	1	1	1	1	1
Couple	0.4	0.2	0.6	1	1	1
Crowd	0	0.4	0.8	1	1	1
Baboon	1	1	1	1	1	1
Pattern	0.8	1	1	1	1	1
Montage	0.6	0	0	1	1	0.2

Table 12 AFES for various bits/pixel

Algorithm	*CDE*			*KBDE*		
bpp	*0.75*	*1*	*1.5*	*0.75*	*1*	*1.5*
Lena	79.4	94	96	17.5	27.3	47
Camera Man	73	87.4	91	11.5	21.3	51.5
Barbara	62.6	51.8	75	14	12	39
Clock	89	-	-	22.5	45.5	98
Bridge	67.2	60.4	84.6	36	19	30
Couple	96.5	97	94.2	24.5	30.5	39.5
Crowd	-	95	99.4	39	26	51
Baboon	60.8	59.8	63	21	14.5	22.5
Pattern	79	69	53.8	20	21.5	24
Montage	94.3	-	-	21	52	99

Even though the above empirical analysis confirms that KBDE performs better than CDE, it is necessary to confirm the results statistically. Hence, one tailed t-test (hypothesis testing) is used to compare the performance of both the algorithms. As a null hypothesis, H_0 is assumed that there is no significant difference between the KBDE and CDE, whereas the alternative hypothesis H_1 is that KBDE is more efficient than CDE at the 5% significance level.

Table 13 Successful performance for various bits/pixel

Algorithm	*CDE*			*KBDE*		
bpp	*0.75*	*1*	*1.5*	*0.75*	*1*	*1.5*
Lena	79.4	105	101	17.5	27.3	47
Camera Man	73	87.4	91	11.5	21.3	51.5
Barbara	62.6	51.8	75	14	12	39
Clock	445	-	-	22.5	45.5	490
Bridge	67.2	60.4	84.6	36	19	30
Couple	241.25	485	157	24.5	30.5	39.5
Crowd	-	237.5	124.25	39	26	51
Baboon	60.8	59.8	63	21	14.5	22.5
Pattern	98.75	69	53.8	20	21.5	24
Montage	157.22	-	-	21	52	495

Table 14 T-test results for average of performance measures

Measures	*P value in t test*		*Significance level*
	After 50 generations	*After 100 generations*	
Average unfitness value	0.0008	0.0035	0.05
Average best of generations	0.0050	0.0015	0.05
Optimisation accuracy	0.0127	0.0076	0.05
Likelihood of evolution leap	0.0142	0.0102	0.05

One tailed t-test with 0.05 as level of significance (α) is applied on the performance measures such as average best unfitness value, average best of generations, optimisation accuracy and likelihood of evolution leap and their obtained p-value are shown in Table 14. If the p-value is less than α, then the null hypothesis is rejected else it is not rejected. From Table 14, it is observed that p-value of all performance measures is less than 0.05 which indicates the rejection of the null hypothesis H_0. Therefore, the statistical results confirm that KBDE is more efficient than CDE with a confidence level of 90%.

6 Conclusions

In this paper, a KBDE algorithm has been proposed to search the optimal quantisation table for the JPEG baseline algorithm. Image characteristics and knowledge about image compressibility are incorporated in CDE operators which provide a better initial population, variable selection pressure and enhanced search capability to accelerate the evolution search in the feasible region. KBDE is able to produce the similar results as CDE in terms of MSE and PSNR with 29% improved convergence rate. Also an extensive comparative analysis has been made between KBDE and CDE in terms of their accuracy, search capability, convergence speed and reliability. The analysis report shows that KBDE guarantees a feasible solution in a less number of generations. Also the empirical results are confirmed by statistical hypothesis test (t-test). Possible direction for the future work includes the use of surrogates in fitness approximation to decrease the computational time further.

References

Ali, W.M., Hegazy, Z.S. and Abd-Elaziz, T. (2013) 'Real parameter optimization by an effective differential evolution algorithm', *Egyptian Informatics Journal*, Vol. 14, No. 1, pp.37–53.

Andrew, M.S., Monte, L. and Whitley, L.D. (2007) 'Differential evolution and non-separability: using selective pressure to focus search', *GECCO 2007 Genetic and Evolutionary Computation Conference*, GECCO, London, pp.1428–1435.

Ankush, M., Aveek, K.D., Prithwijit, M., Das, S. and Suganthan, P.N. (2011) 'Modified differential evolution with local search algorithm for real world optimization', *IEEE IEEE 2011Evolutionary Computation Congress*, IEEE, USA, pp.1565–1572.

Asafuddoula, M., Ray, T. and Sarker, R. (2011) 'An adaptive differential evolution algorithm and its performance on real world optimization problems', *IEEE 2011 Evolutionary Computation Congress*, IEEE, USA, pp.1057–1062.

Babu, B.V. and Rakesh, A. (2006) 'Modified differential evolution (MDE) for optimization of non-linear chemical processes', *Computers and Chemical Engineering*, Vol. 30, No. 7, pp.989–1002.

Chen, C.H. and Yang, S.Y. (2013) 'A knowledge-based cooperative differential evolution for neural fuzzy inference systems', *Soft Comput.*, Vol. 17, No. 5, pp.883–895.

Chen, C.H. and Yang, S.Y. (2014) 'Neural fuzzy inference systems with knowledge-based cultural differential evolution for nonlinear system control', *Inform Sciences*, Vol. 270, No. 1, pp.154–171.

Ching-Hung, L., Che-Ting, K. and Hao-Han, C. (2012) 'Performance enhancement of the differenetial evolution algorithm using local search and a self-adaptive scaling factor', *International Journal of Innovative Computing, Information and Control*, Vol. 8, No. 4, pp.2665–2679.

Daniela, Z. (2009) 'Influence of crossover on the behavior of differential evolution algorithms', *Appl. Soft. Comput.*, Vol. 9, No. 3, pp.1126–1138.

Das, S., Abraham, A., Chakraborty, U.K. and Amit, K. (2009) 'Differential evolution using a neighborhood-based mutation operator', *IEEE Transactions on Evolutionary Computation*, Vol. 13, No. 3, pp.526–553.

Islam, S.M., Das, S., Ghosh, S., Roy, S. and Suganthan, P.N. (2012) 'An adaptive differential evolution algorithm with novel mutation and crossover strategies for global numerical optimization', *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, Vol. 42, No. 2, pp.482–500.

Kumar, P., Pant, M. and Abraham, A. (2011) 'Two enhanced differential evolution variants for solving global optimization problems', *IEEE 2011 Nature and Biologically Inspired Computing World Congress*, IEEE, USA, pp.201–206.

Lazzerini, B., Marcelloni, F. and Vecchio, M. (2010) 'A multi-objective evolutionary approach to image quality/compression trade-off in JPEG baseline algorithm', *Appl. Soft. Comput.*, Vol. 10, No. 2, pp.548–561.

Ma, H. and Zhang, Q. (2012) 'Research on cultural-based multi-objective particle swarm optimization in image compression quality assessment', *Optik-Internationl Journal for Light and Electon. Optics.*, Vol. 124, No. 10, pp.957–961.

Mashwani, W.K. (2014) 'Enhanced versions of differential evolution: state-of-the-art survey', *Int. J. of Computing Science and Mathematics*, Vol. 5, No. 2, pp.107–126.

Milan, T. and Nebojsa, B. (2014) 'JPEG quantization tables selection by the firefly algorithm', *IEEE 2014 Image Processing Conference*, IEEE, USA, p.376.

Sandeep, K., Vivek, K.S. and Rajani, K. (2014) 'Memetic search in differential evolution algorithm', *International Journal of Computer Applications*, Vol. 90, No. 6, pp.40–47.

Sherlock, B.G., Nagpal, A. and Monro, D.M. (1994) 'A model for JPEG quantization, *International Symposium on Speech, Image Processing and Neural Networks, ISSIPNN'94*, DOI 10.1109/SIPNN.1994-344938.

Storn, R. and Price, K.V. (1995) *Differential Evolution: A Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces*, Tech. Rep. TR-95-012, ICSI, USA.

Vinoth Kumar, B and Karpagam, M. (2014) 'Knowledge based genetic algorithm approach to quantization table generation for the JPEG baseline algorithm', *Turk. J. Electr. Eng. Co.*, DOI: 10.3906/elk-1310-179 [online] http://mistug.tubitak.gov.tr/bdyim/kabul.php?dergi=elk (accessed 28 February 2015).

Vinoth Kumar, B. and Karpagam, M. (2015) 'Differential evolution versus genetic algorithm in optimising the quantisation table for JPEG baseline algorithm', *Int. J. Advanced Intelligence Paradigms*, Vol. 7, No. 2, pp.111–135.

Vinoth Kumar, B., Karpagam, G.R. and Vijaya Rekha, N. (2015) 'Performance analysis of deterministic centroid initialization method for partitional algorithms in image block clustering', *Indian Journal of Science and Technology*, Vol. 8, No. S7, pp.63–73.

Wallace, G. (1992) 'The JPEG still picture compression standard', *IEEE Trans. Consum. Electron.*, Vol. 38, No. 1, pp.18–34.

Wenyin, G., Zhihua, C., Charles, X.L. and Hui, L. (2011) 'Enhanced differential evolution with adaptive strategies for numerical optimization', *IEEE Transcations on Systems, Man, and Cybernetics, Part B: Cybernetics*, Vol. 41, No. 2, pp.397–413.

Wu, Y.G. (2004) 'GA-based DCT quantization table design procedure for medical images', *IEE Proceedings Vision, Image and Signal Processing*, Vol. 151, No. 5, pp.353–359.

Xuemei, Y. (2010) 'Differential evolution with a new mutation operator for solving high dimensional continuous optimization problems', *Journal of Computational Information Systems*, Vol. 6, No. 9, pp.3033–3039.

Zhenyu, Y., Ke, T. and Xin, Y. (2008) 'Self-adaptive differential evolution with neighborhood search', *IEEE 2008 Evolutionary Computation Congress*, IEEE, USA, pp.1110–1116.