Using Genetic Algorithms to Find New Color Transformations for JPEG-2000 Image Compression

Mohammed S. Al-Rawi, Abdel-Latif Abu-Dalhoum, Yousef Salah, Wesam Al-Mobaideen, Ansar Khoury
Computer Science Department
King Abdullah II School for Information Technology, The University of Jordan
Amman 11942, Jordan
rawi@ju.edu.jo*, rawi707@yahoo.com, a.latif@ju.edu.jo, yousef_sal@yahoo.com, wesmoba@ju.edu.jo, ansar@ju.edu.jo

Abstract: Image compression techniques play an important role in multimedia applications. The JPEG-2000 which is based on the wavelet transform is a promising image compression technique expected to replace the current discrete cosine transform based compression known as JPEG. In this paper, genetic algorithms are used to optimize the coefficients of the RGB to YC_bC_r color transform used in JPEG-2000 and to find alternate color transforms. Color transformation in JPEG-2000 is an early phase of the compression process intended to improve the compression performance. The matrix elements are optimized using fitness functions based on the root mean square error between the original and the reconstructed image. The resultant color transformations revealed an enhancement in the JPEG-2000 codec by.

Key-Words: - Genetic Algorithms, Irreversible Color Transform, Reversible Color Transform, JPEG2000, Image Compression, Wavelet Transform, ITU-R, jasper.

1 Introduction

JPEG image compression [1, 2] which is based on the discrete cosine transform is a powerful tool used in multi-media applications. A step beyond JPEG is the JPEG-2000 that is based on wavelet transform [3] which is one of the most promising image compression methods. Its superiority in achieving low bit rate compression, error resilience, and other features promotes it to become the tomorrow's compression standard and leads to the JPEG-2000 ISO. As referred to the JPEG abbreviation which stands for Joint Photographic Expert Group, JPEG-2000 codec is more efficient than its predecessor JPEG and overcomes its drawbacks [4]. It also offers higher flexibility compared to even many other codec such as region of interest, high dynamic range of intensity values, multi component, lossy and lossless compression, efficient computation, compression rate control, etc. The robustness of JPEG-2000 stems from its utilization to the Discrete Wavelet Transform (DWT) in encoding the image data. DWT exhibits high effectiveness in image compression due to its support to multiresolution representation in both spatial and frequency domains. In addition, DWT supports progressive image transmission and region of interest coding [5].

The general structure of the JPEG-2000 standard codec is shown in Fig. 1 where the whole compression system is divided into three phases; image preprocessing, compression, and compressed bitstream formation [6]. Image preprocessing involves three operations; tiling, DC level shifting, and multicomponent transformation [7]. Tiling is the process of portioning large images into rectangle nonoverlapping blocks called tiles. Tiling offers flexibility in memory handling and reduces the complexity of the DWT. DC level shifting shifts the image data so as its dynamic range is approximately centered near zero. Then, the multi-component color transform is performed on the color components of the image. The compression phase of the JPEG-2000 is embodied in performing the DWT for the image components, quantizing the resulted coefficients, and entropy encoding the quantized data using bit plane arithmetic coding. There are two DWT methods supported in JPEG-2000, irreversible DWT and reversible DWT. The former is applied in the case of lossy compression and uses the 9/7 wavelet filter and the latter is applied in lossless compression uses the 5/3 wavelet filter [5].

In order to eliminate the correlation between different components, the Color Transformation (*CT*) is used to map the color space of an image which is consisted of three components for an RGB image to another color space. For example, the *CT* used in the standard JPEG-2000 maps the RGB color space to YC_bC_r which is based on the CCIR recommendation 601-1 recently known as ITU-R [8]. Moreover, two *CT* versions are used in JPEG-2000, the Reversible Color Transformation (*RCT*) used for lossless compression and Irreversible Color Transformation (*ICT*) that may be used in lossy and lossless compression. In JPEG-2000, both *ICT* and *RCT* are fixed and used for all types of images without any regard to the color distribution of that image. In this paper, we propose a method to find new *CT*s with the hope to improve the performance of the current *ICT* and *RCT* adopted for JPEG-2000. To do the comparison between different *CT*s, a model is developed to calculate the performance of the compression algorithms.

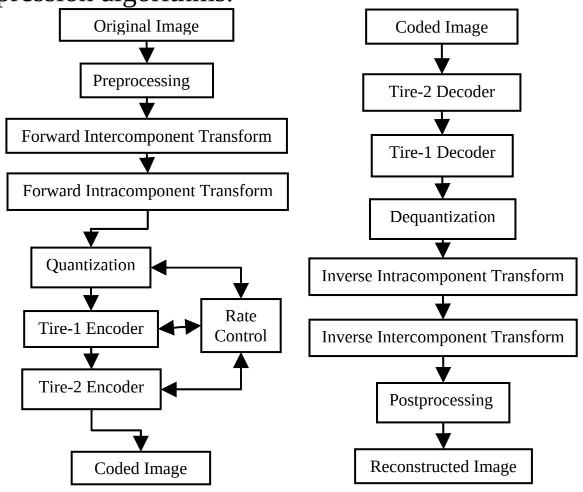

Fig.1: JPEG-2000 codec structure, the left part shows *Coder* while the right part shows the *Decoder* [12].

Genetic algorithms (GAs) are evolutionary algorithms that provide powerful and robust means of solving optimization problems. Inspired by evolution in biological systems, John Holland [9] developed GAs to simulate evolution of different communities as well as solving optimization problems. They are efficient in solving hard and intractable problems where the search space is huge and the mathematical optimization is infeasible. To enlist previous experimentations of image compression methods via GAs, Parent and Nowe in [10] presented a new approach to implement genetic programming to lossless data compression. The evolved programs were preprocessors aimed to promote compression rate of a given data. Since GAs may take considerable computation time, the authors demystified how their approach lessens the computation time of compression. In another work done by Vences and Rudomin in [11] where they present a method that uses genetic algorithms to find a Local Iterated Function System (LIFS) that encodes a single image. It is obvious in the literature that the mentioned methods use GAs to compress the image each time one needs to do image compression. They did not use GAs to optimize some parameters of a compression algorithm as we intend in this work. In fact, we shall use the JPEG-2000 parameters according to the ISO standard, change the compression rate in order to perform GA search for new *CT*s. The new *CT*s should be used for JPEG-2000 image compression without needing GA runs.

The rest of this paper is organized as follow: Section 2 elucidates the intercomponent *CT* used in JPEG-2000. The linking between GAs and the *CT* optimization of JPEG-2000 is the topic of section 3. Section 4 demonstrates the Experimental Results, and section 5 wraps up by the conclusions.

2 Intercomponent Color Transform in JPEG-2000

The JPEG-2000 standard consists of many parts. Part-1 which is the core coding system [3] was standardized and became an international standard. In JPEG-2000, colored images are multicomponent images where each component is partitioned into rectangle, non-overlapping regions called tiles that are compressed individually. Prior to tiling, the multicomponent transformation is applied to the three components of the colored image. A tile is compressed in four stages. First, wavelet transform is conducted that results in subbands of wavelet coefficients. Lossless compression uses an integer-to-integer reversible wavelet transform denoted as the 5/3 wavelet transform, while lossy compression uses a real-to-real irreversible wavelet transform denoted as the 9/7 wavelet transform. Second, the wavelet coefficients are quantized if the target rate is specified. Third, wavelet coefficients are encoded using arithmetic coding. Forth, the bitstream is constructed [7].

The multicomponent transformation is the preprocessing stage by which the JPEG-2000 system diminishes correlation among image components. This offers little redundancy and increased compression performance. The JPEG-2000 Part-1 standard introduces two types of image multicomponent transformation: the *RCT* that can be applied for both lossy and lossless compression, and the *ICT* which is conducted in the case of lossy compression [3]. The *RCT* enables pixels to be recovered back totally when doing image encoding by applying the inverse *RCT*. This integrity in the recovery process stems from the integer-to-integer nature of the *RCT*. The forward *RCT* transform is given by [3]:

$$\begin{aligned}
V_0(x,y) &= \left\lfloor \frac{1}{4}\left(U_0(x,y)+2U_1(x,y)+U_2(x,y)\right)\right\rfloor \\
V_1(x,y) &= U_2(x,y)-U_1(x,y) \\
V_2(x,y) &= U_0(x,y)-U_1(x,y)
\end{aligned}\quad , \qquad (1)$$

where $U_0(x,y)$, $U_1(x,y)$, and $U_2(x,y)$ are the input components that denotes the red, green, and blue color components respectively, $V_0(x,y)$, $V_1(x,y)$, and $V_2(x,y)$ are the resulting transform components and denotes the Y, C_r, and C_b components respectively, $\lfloor w \rfloor$ is the largest integer less than or equal to *w*. The inverse *RCT* which is needed when decompressing the image is given by:

$$\begin{aligned}
U_1(x,y) &= V_0(x,y)-\left\lfloor \frac{1}{4}\left(V_1(x,y)+V_2(x,y)\right)\right\rfloor \\
U_0(x,y) &= V_2(x,y)+U_1(x,y) \\
U_2(x,y) &= V_1(x,y)+U_1(x,y)
\end{aligned}\quad . \qquad (2)$$

The *ICT*, on the other hand, could be used in the case of lossy compression. It throws some information in the original image components by using fractional coefficients in the transformation matrix. So, pixels are not recovered back precisely when the image is decoded. The forward *ICT* [3] is given by:

$$\begin{bmatrix} V_0(x,y) \\ V_1(x,y) \\ V_2(x,y) \end{bmatrix} = \begin{bmatrix} 0.29900 & 0.58700 & 0.11400 \\ -0.16875 & -0.33126 & 0.50000 \\ 0.50000 & -0.41869 & -0.08131 \end{bmatrix} \begin{bmatrix} U_0(x,y) \\ U_1(x,y) \\ U_2(x,y) \end{bmatrix}, \quad (3)$$

while the inverse *ICT* is given by:

$$\begin{bmatrix} U_0(x,y) \\ U_1(x,y) \\ U_2(x,y) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1.40200 \\ 1 & -0.34413 & -0.71414 \\ 1 & 1.77200 & 0 \end{bmatrix} \begin{bmatrix} V_0(x,y) \\ V_1(x,y) \\ V_2(x,y) \end{bmatrix}. \quad (4)$$

We will use the *ICT* and *RCT* in performance analysis and to seed the population of the GA engine in some experiments.

3 Genetic Algorithms and JPEG-2000 Color Transformation search

We are using GAs to search for better *CT*s applied to JPEG-2000. The purpose of the new *CT*s is to increase the performance of JPEG-2000. The following sections describe the steps needed to achieve this goal.

3.1 GAs in Search and Optimization

As GAs are inspired from biological evolution, the first step is the encoding of possible solutions as values called chromosomes. GA starts by an initial population of those chromosomes that are initiated randomly or explicitly. Generations are simulated by mating the fittest individuals in the populations and applying some biological inspired events on the resulted offspring. Mutation and Crossover are two salient events that help us creep in the search space looking for the global optimal value that is being sought or a near value to it. The fitness function is a significant issue in the scope of GAs to find the fittest individuals that are adopted to constitute the next generation. The fitness function is a problem oriented aspect, and it minimizes or maximizes the search toward the fittest solution.

3.2 The Chromosome Encoding Structure

Since the *CT* matrix is a signed real-valued matrix, the real value encoding is used for chromosomes representation. Since the *CT* is a 3×3 matrix, the chromosome should contain nine genes. The *CT* matrix and its chromosome encoding that is used in this work is given in Equation (5) and (6).

$$CT = \begin{bmatrix} c_{1,1} & c_{1,2} & c_{1,3} \\ c_{2,1} & c_{2,2} & c_{2,3} \\ c_{3,1} & c_{3,2} & c_{3,3} \end{bmatrix}, \quad (5)$$

$$CT\_Chromosome = [c_{1,1}, c_{2,1}, c_{3,1}, c_{1,2}, c_{2,2}, c_{3,2}, c_{1,3}, c_{2,3}, c_{3,3}]. \quad (6)$$

It must be noted that the encoding shown above is performed via a double vector and not a binary string. For more information on how to use double vectors as GA chromosomes the reader is referred to [12].

3.3 The Fitness Function(s)

Three fitness functions are applied to assess the effectiveness of each GA generated *ICT* matrix which is denoted as *GA-ICT*. Each fitness function is formulated based on the theoretical and mathematical knowledge of the red, green, blue Root Mean Squared Error (RMSE) [13, 14] values of a colored image. Let *f* be an *M*×*N* image and *f'* is the corresponding reconstructed image after compressing and decompressing of image *f*, then the RMSE is given by:

$$e = \sqrt{\frac{1}{MN}\sum_{x=1}^{M}\sum_{y=1}^{N}[f(x,y) - f'(x,y)]^2}, \quad (7)$$

where e is the RMSE value. In this work, e_R, e_G, and e_B denote the RMSE for the red, green and blue component of the image respectively. The RMSE shown in (7) cannot be directly used as a fitness function. Intuitively, the fitness should measure the possible RMSE over all possible compression rates. A strategy that best describes the entire compression rates for a specific chromosome is to calculate the integral of the receiver operating curve (ROC) that measures the change in RMSE when the compression rate is changed. In doing this we are calculating the area under the ROC (AUROC) therefore the minimum the AUROC refers to the best *CT_chromosome*. Lower AUROC value indicates better image fidelity over a wide range of compression rates. The following steps describe how to calculate the fitness function for one individual chromosome:

1. The genes of the current chromosomes are passed as vector to the JPEG-2000 color transform function inside the transcoder implementation.
2. If the matrix is singular or it is ill-conditioned, a very high fitness penalty value is assigned so that the current chromosome is discarded.
3. After loading the image that is used in the *ICT* optimization of the current chromosome, it is encoded into the JPEG-2000 image file stream at the specified compression rate.
4. If the encoding step fails to produce the JPEG-2000 stream due to errors in fulfilling the JPEG-2000 color transform criteria a very high fitness value is assigned to the current *CT_Chromosome* as a penalty.
5. The image is decoded back with the inverse *ICT* matrix of the current *CT_Chromosome*.
6. Using equations (8) or (9), or (10), the error is computed for the original image and the reconstructed image. Let $f_{ICT}(\chi)$ be the total compromise error obtained by using a certain *CT_Chromosome* with a compression rate χ, given by:

$$g_{CT}^{(1)}(\chi)=\frac{|e_R-e_G|+|e_R-e_B|+|e_G-e_B|+(e_R*e_G*e_B)}{e_R+e_G+e_B}, \quad (8)$$

and in a similar manner other metrics are proposed as:

$$g_{CT}^{(2)}(\chi)=e_R*e_G*e_B, \quad (9)$$

$$g_{CT}^{(3)}(\chi)=\frac{e_R+e_G+e_B}{3}. \quad (10)$$

7. The compression rate is then increased gradually.
8. The process is repeated by returning back to step 3 until the rate reaches a certain threshold and the resultant RMSEs versus compression rates are stored in a vector.
9. The fitness is calculated by finding the AUROC value for the above compromise error as follows:

$$Fitness1=\int_{\chi_{\text{Start}}}^{\chi_{\text{threshold}}} g_{CT}^{(1)}(\chi)\,d\chi, \quad (11)$$

$$Fitness2=\int_{\chi_{\text{Start}}}^{\chi_{\text{threshold}}} g_{CT}^{(2)}(\chi)\,d\chi, \quad (12)$$

$$Fitness3=\int_{\chi_{\text{Start}}}^{\chi_{\text{threshold}}} g_{CT}^{(3)}(\chi)\,d\chi, \quad (13)$$

where χ_{Start} denotes the start of the compression rate, $\chi_{\text{threshold}}$ denotes the last compression rate used in the fitness function. The lower the fitness, the better the compression over many compression rates is.

3.4 *Determining* $\chi_{\text{Threshold}}$

In this work, two approaches are used to determine the $\chi_{\text{Threshold}}$. In the first approach, $\chi_{\text{Threshold-1}}$ is estimated by performing lossless compression for the image under consideration. Since lossless compression does not specify the compression rate, the resulted encoded image stream size will indicate the suitable compression rate that guarantees no loss of information.

Table 1: Compression rate thresholds of JPEG2000

Image	$\chi_{\text{threshold-1}}$	$\chi_{\text{threshold-2}}$
Fractal	0.517	0.20
Lena	0.565	0.28
Peppers	0.617	0.33
Baboon	0.753	0.50
Texture	0.759	0.45

In the second approach, $\chi_{\text{Threshold-2}}$ is estimated by doing lossy compression for each image over the entire range of compression rates. The value of this threshold is specified by the point where the error of the three image components values saturates. Table 1 shows the estimated value of $\chi_{\text{Threshold-1}}$ and $\chi_{\text{Threshold-2}}$ for each image. The images shown in Table 1 are the five images used in the experiments performed in this work. For more clarity, the full implementation is shown in Algorithm 1.

4 Experimental Results

JasPer [12] is an open source JPEG-2000 transcoder toolkit. The ISO/IEC N2415 standard has adopted JasPer to be one of the standardized JPEG-2000 implementation. JasPer software is developed by Michael Adams and written completely in the C programming language. Our interest of JasPer embodied in its support to the JPEG-2000 Part-1 codec. In this work, GA optimization engine is linked with the JasPer 1.701.0 software in order to search for better *CT*s. MATLAB has been used to perform the GA operations, fitness calculation, *jasper.exe* invocation that performs JPEG-2000 codec. To perform the GA search for *CT*s and the performance analysis of the results, five testing images have been used. As shown in Fig. 2, the five bitmap images show variant color distributions and vast spatial content.

Every GA experiment searches for a *CT*. The obtained *CT* is called *ICT* since the chance of getting an *RCT* is very small and the obtained *CT* is irreversible. We also refer to the *ICT* resultant form GA optimization as *GA-ICT*. Table 2 shows the GA parameters that are applied during the optimization process of this work. Several experiments have been performed and those with the best performance (highest resultant fitness) are enlisted in this paper. The experiments are performed for each of the testing image using each fitness function. Each experiment outputs a *GA-ICT* as shown in Table 3`and for each *ICT* matrix, the inverse *ICT* can be simply found as matrix inverse of the corresponding *ICT*.

Though each *GA-ICT* is found from one image, its performance has been calculated as the average AUROC and the standard deviation over all components of the five images. It is obvious that *ICT-9* outperforms all other *ICTs* including those of the JPEG2000 standard. The following are the best average AUROC values, *ICT-9* has the minimum average (rank 1) which is 2.12548 and its standard deviation is 1.06158. *ICT-10* has a rank 2 average which is 2.1542 and a rank 1 standard deviation given by 1.03203. *ICT*-4 comes third with 2.1803 as its average value. *ICT*-5 comes in the fourth place with an average given by 2.187. *ICT*-15 comes in the fifth place with rank 5 an average given by 2.19207. It is astonishing that the top five resulted *ICT's* are all obtained by optimizing GA using the texture or the fractal image. This is expected since the texture and the fractal have vast spatial contents.

Algorithm 1: GA optimizing color transformation of JPEG-2000

```
1:    iteration ← 0, N← No Of individuals in the
2:    population
3:    create the initial population $P_N(0)$
4:    evaluate $P_N(0)$ using Fitness1
5:    do
6:      iteration ← iteration+1
7:      CT_Chromosome← select individuals from
8:      $P_N(iteration)$ for reproduction
9:       apply crossover and/or mutation operations on
10:     CT_Chromosome
11:      generate the new population $P_N(iteration)$
12:      evaluate $P_N(iteration)$ using Fitness1
13:   until termination condition is met
      return best individual of $P_N(iteration)$
```

Fig. 2: Testing images used with their histograms. Left to right, top to bottom shows, Lena (512x512), Babbon (512x512), Peppers (512x512), Fractal (800x584), and Texture (400x400) all as 24 bit-per-pixel.

Table 2: GA parameters during each experiment

GA Parameter	**Value**
Population Initial Range	[-2, +2]
Chromosome Length	9
Population Size	30
Elite Count	1
Crossover Fraction	0.80
Total Generations Allowed	200
Selection	Rank Selection
Mutation Function	Gaussian
All Stall limits	Infinite

The estimated average AUROC for all testing images when applying each of the *GA-ICT*'s of Table 3 is demonstrated in Fig. 3. A comparison with JPEG-2000 has been performed without any *CT* (*NO-CT*), the *RCT*, and the *ICT* of the JPEG-2000 ISO (*Def-ICT*) is also performed. In all the experiments shown in Fig. 3, the AUROC is calculated within the range $\chi_{\text{Start}} = 0.01$ to $\chi_{\text{threshold}} = 0.5$.

Table 3: *GA-ICT*s obtained via several GA experiments.

ID	**Fitness Function**	**Used Image**	**Name of *GA-ICT***	***GA-ICT* found**
1	Fitness1	Lenn	*ICT-1*	$\begin{bmatrix} -1.33122 & 0.632144 & 0.114 \\ -0.16875 & -0.89546 & 1.236005 \\ 0.892269 & 1.858684 & 0.772574 \end{bmatrix}$
2	Fitness1	Baboon	*ICT-2*	$\begin{bmatrix} -1.98337 & 0.421759 & 1.309401 \\ 0.869628 & -1.64245 & 1.067539 \\ 1.62872 & 1.950096 & 1.010238 \end{bmatrix}$
3	Fitness1	Peppers	*ICT-3*	$\begin{bmatrix} 2.487705 & -0.84213 & -0.02721 \\ -0.26607 & -1.01234 & 2.203384 \\ -0.4681 & -2.57219 & -1.24737 \end{bmatrix}$
4	Fitness1	Fractal	*ICT-4*	$\begin{bmatrix} 0.731309 & 0.437299 & 0.700563 \\ -0.38607 & -0.6296 & 1.034513 \\ 0.907546 & -1.1122 & 0.210193 \end{bmatrix}$
5	Fitness1	Texture	*ICT-5*	$\begin{bmatrix} 0.908202 & 1.709937 & 0.98824 \\ 0.041652 & -1.2303 & 1.325266 \\ 1.61251 & -1.32556 & -0.16492 \end{bmatrix}$
6	Fitness2	Lenna	*ICT-6*	$\begin{bmatrix} -2.58238 & 0.343586 & 1.775097 \\ 1.300281 & -2.11495 & 1.504676 \\ 2.417481 & 2.241595 & 1.270104 \end{bmatrix}$
7	Fitness2	Baboon	*ICT-7*	$\begin{bmatrix} 0.952829 & 2.469947 & 0.491108 \\ 0.045988 & -1.32076 & 1.664826 \\ 2.578607 & -1.6051 & -0.4855 \end{bmatrix}$
8	Fitness2	Peppers	*ICT-8*	$\begin{bmatrix} 1.365613 & 0.315012 & -0.19562 \\ 0.114701 & -0.07535 & -1.37662 \\ 0.511523 & -1.44096 & 0.306857 \end{bmatrix}$
9	Fitness2	Fractal	*ICT-9*	$\begin{bmatrix} -1.63422 & -1.29789 & -1.04512 \\ -0.64495 & 1.626064 & -0.9845 \\ -2.39129 & 0.805414 & 1.589638 \end{bmatrix}$
10	Fitness2	Texture	*ICT-10*	$\begin{bmatrix} 0.207585 & 0.92804 & -1.17135 \\ -1.17023 & 0.782119 & 0.410022 \\ -1.08872 & -0.75713 & -0.82435 \end{bmatrix}$
11	Fitness3	Lena	*ICT-11*	$\begin{bmatrix} -0.57443 & 0.990345 & -0.76308 \\ -1.25854 & -0.72338 & -0.3815 \\ 1.15213 & -0.22393 & -0.95791 \end{bmatrix}$
12	Fitness3	Baboon	*ICT-12*	$\begin{bmatrix} 0.650112 & 0.852781 & 0.46132 \\ -0.69284 & -0.12029 & 0.728175 \\ 0.687827 & -0.92932 & 0.46694 \end{bmatrix}$
13	Fitness3	Peppers	*ICT-13*	$\begin{bmatrix} 0.299 & 1.080893 & 0.286585 \\ -0.16875 & -0.36301 & 0.833695 \\ 1.033431 & -0.51652 & 0.041639 \end{bmatrix}$
14	Fitness3	Fractal	*IC-14*	$\begin{bmatrix} -0.44342 & 2.265336 & -1.87188 \\ 1.48281 & 0.863432 & 1.18756 \\ 2.344087 & -2.06339 & -0.32961 \end{bmatrix}$
15	Fitness3	Texture	*ICT-15*	$\begin{bmatrix} 0.538221 & 0.720023 & 0.546425 \\ 0.268671 & -0.84911 & 0.609377 \\ -0.898 & 0.205087 & 0.697049 \end{bmatrix}$

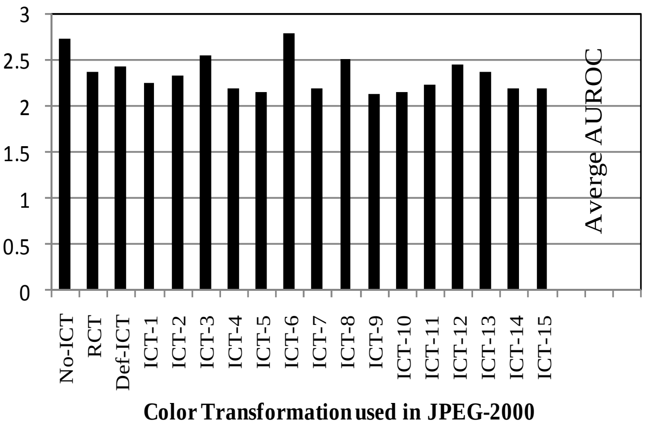

Fig. 3: Showing the average area under ROC between the image and the reconstructed image versus the color transform used in JPEG-2000. *ICT-9* is the best color transform.

It is important to show the average and standard deviation at each color component[15, 16] in order to inspect the performance at each color component. It is obvious that *ICT-9* outperforms all other *ICT's* including those of the JPEG2000 standard.

A good measure that can be used to estimate the performance of the resultant *GA-ICT* is its inversion property. An $n \times n$ matrix A is called invertible (non-singular) if there exist an $n \times n$ matrix B that holds the condition $AB = BA = I_n$ where I_n denotes the $n \times n$ identity matrix. A singular matrix is prohibited in our experimentations due to the need of the matrix inverse at the decoder phase. On the other hand, *GA-ICT's* should also be far from singularity. In numerical calculations, matrices which are invertible, but close to a non-invertible matrix, can still be problematic; such matrices are said to be ill-conditioned. A matrix with a low condition number is said to be well-conditioned, while a matrix with a high condition number is said to be ill-conditioned. The condition number of a matrix designates the bound of accuracy of the matrix that satisfies a specific solution.

Table 4: AUROC analysis for all the GA-ICT matrices

Image\ Component	**Image**	**AUROC using the specified Color Transform**								
		NO-CT	**RCT**	**Def-ICT**	***ICT-1***	***ICT-2***	***ICT-3***	***ICT-4***	***ICT-5***	***ICT-6***
RED	**Lenna**	1.09801	1.27704	1.18514	1.02605	1.01037	0.98732	1.00060	1.07697	0.77414
	Baboon	3.57120	3.61609	3.52015	3.27226	3.24616	3.31880	3.13325	3.31010	4.03875
	Peppers	1.43894	1.74953	1.77235	1.48045	1.46089	1.37907	1.50331	1.52821	2.19640
	Fractal	2.89037	1.77853	1.78012	2.10034	1.88835	2.60489	1.57624	1.65209	2.24440
	Texture	4.34162	4.40811	4.30407	4.18419	4.01277	4.96136	3.88803	4.07748	4.96137
	Average	2.66803	2.56586	2.51237	2.41266	2.32371	2.65029	2.22029	2.32897	2.84301
	Std Dev	1.38218	1.36420	1.32958	1.30062	1.26122	1.59464	1.22794	1.29294	1.65634
Green	**Lenna**	1.15924	0.99151	1.16270	0.86847	1.17084	1.14422	1.11649	0.85398	0.98165
	Baboon	3.49832	2.67340	2.82948	2.57330	3.44472	3.53361	3.07692	2.42786	3.30777
	Peppers	1.36793	1.26788	1.37565	1.14220	1.51879	1.50519	1.40277	1.12192	1.83394
	Fractal	2.73552	1.46685	1.55325	1.71684	1.95063	2.46372	1.55539	1.39216	2.13535
	Texture	4.05023	3.24284	3.29591	3.17646	4.02515	3.94465	3.64371	3.02464	3.94465
	Average	2.56225	1.92850	2.04340	1.89545	2.42203	2.51828	2.15906	1.76411	2.44067
	Std Dev	1.27624	0.97594	0.95505	0.96882	1.24694	1.22299	1.12583	0.92312	1.18363
Blue	**Lenna**	1.32872	1.41049	1.58803	1.19870	1.11162	1.00102	1.17775	1.22173	1.17556
	Baboon	4.03999	3.78326	3.99798	3.50150	3.21172	2.97228	3.17477	3.43858	4.43743
	Peppers	1.56073	1.73637	1.82225	1.47956	1.30027	1.21378	1.38107	1.49561	2.27328
	Fractal	3.19280	1.83894	1.93618	1.92476	1.93436	2.19631	1.58752	1.68820	2.52456
	Texture	4.64497	4.26659	4.30732	3.97396	3.61329	4.99537	3.58864	3.90447	4.99537
	Average	2.95345	2.60714	2.73036	2.41570	2.23425	2.47575	2.18195	2.34972	3.08124
	Std Dev	1.47297	1.31504	1.30901	1.24555	1.12692	1.61559	1.11441	1.22906	1.58882
Total Average		2.72791	2.36716	2.42871	2.24127	2.32666	2.54811	2.18710	2.14760	2.78831
Total Std		1.24485	1.14365	1.11970	1.08393	1.08781	1.33361	1.03533	1.07212	1.35950

	Image	AUROC using the specified Color Transform								
		ICT-7	*ICT-8*	*ICT-9*	*ICT-10*	*ICT-11*	*ICT-12*	*ICT-13*	*ICT-14*	*ICT-15*
RED	**Lenna**	0.90719	0.99452	0.80151	0.89785	0.76833	1.01753	1.05341	0.94127	1.01501
	Baboon	2.90878	3.17666	2.58962	2.89907	2.45066	3.24080	3.32502	3.00387	3.11841
	Peppers	1.34510	1.34074	1.22636	1.33874	1.21172	1.50487	1.49152	1.44002	1.51580
	Fractal	1.72158	2.43087	1.41357	1.51780	1.55333	2.00136	2.26764	1.51691	1.60185
	Texture	3.76593	4.04061	3.43294	3.59955	3.28434	4.08093	4.23482	3.84598	3.98081
	Average	2.12971	2.39668	1.89280	2.05060	1.85368	2.36910	2.47448	2.14961	2.24637
	Std Dev	1.17908	1.26423	1.08654	1.14440	1.01038	1.26482	1.30869	1.22182	1.24841
Green	**Lenna**	0.82903	1.20952	1.05061	1.08486	1.09408	1.23851	0.94451	0.97939	1.05204
	Baboon	2.41681	3.70185	3.00240	3.08082	3.17904	3.62825	2.72572	2.79172	2.84618
	Peppers	1.08868	1.44303	1.39905	1.40828	1.50515	1.58583	1.18500	1.29589	1.33412
	Fractal	1.51089	2.74476	1.48992	1.51913	1.90755	2.30814	1.88923	1.44238	1.49661
	Texture	3.01719	4.32580	3.71005	3.63224	3.87917	4.27901	3.28627	3.47396	3.45145
	Average	1.77252	2.68499	2.13041	2.14506	2.31300	2.60795	2.00614	1.99667	2.03608
	Std Dev	0.92068	1.36467	1.15831	1.13424	1.17321	1.30800	0.99643	1.07792	1.05020
Blue	**Lenna**	1.32162	1.10302	1.19202	1.21006	1.23721	1.17817	1.31325	1.24105	1.24178
	Baboon	3.84178	3.24370	3.42671	3.37661	3.62744	3.31505	3.67747	3.50489	3.32322
	Peppers	1.64784	1.33052	1.52684	1.42053	1.58714	1.37121	1.55242	1.55738	1.48425
	Fractal	1.97304	2.42031	1.65187	1.64046	2.01893	2.13722	2.38609	1.63581	1.68149
	Texture	4.37992	3.90067	3.96866	3.68704	4.03952	3.77577	4.26807	4.03739	3.73808
	Average	2.63284	2.39965	2.35322	2.26694	2.50205	2.35548	2.63946	2.39531	2.29377
	Std Dev	1.38191	1.20314	1.25352	1.16982	1.25505	1.15548	1.29755	1.27856	1.14921
Total Average		2.17836	2.49377	2.12548	2.15420	2.22291	2.44418	2.37336	2.18053	2.19207
Total STD		1.10930	1.15201	1.06158	1.03203	1.06463	1.11907	1.11463	1.08205	1.03655

Accordingly, the condition number of the *GA-ICT's* is less than the condition number of the *def-ICT*. Does a matrix with smaller condition number indicates a better image reconstruction which implies better image fidelity?. Table 5 lists the condition number of all the *GA-ICT's* and the *def-ICT*. The table shows that most of the *GA-ICT's* have condition a number less than the *def-ICT* which reveals the reason of better image reconstruction of these *GA-ICT's*.

Table 5: The condition number of all the *ICT*'s

Used CT	Condition Number	Used CT	Condition Number
Def- ICT	1.7519	*ICT-8*	1.3215
ICT-1	1.6167	*ICT-9*	1.6179
ICT-2	1.4809	*ICT-10*	1.1792
ICT-3	1.2281	*ICT-11*	1.5686
ICT-4	1.5236	*ICT-12*	1.3403
ICT-5	1.6803	*ICT-13*	1.4423
ICT-6	1.4576	*ICT-14*	1.9242
ICT-7	1.9121	*ICT-15*	1.1635

The results of Table 5 show that *ICT-15* has the least condition number which indicates that the best image decompression might occur when this matrix is employed. This is not enough, however, to judge that *ICT-15* is the best. Going back to Table 4 we see that *ICT-9* has the minimum average (rank 1) which is 2.12548 and the standard deviation is 1.06158, also *ICT-10* has a rank 2 average which is 2.1542 and a rank 1 standard deviation given by 1.03203. *ICT-4 comes third with 2.1803 average*, and *ICT-5* comes in the fourth place with given by 2.187, and *ICT-15* comes in the fifth place with rank 5 average given by 2.19207. It is astonishing that the top five obtained *ICT's* all are obtained by optimizing GA using the texture or the fractal image. This is expected since the texture and the fractal image have uniformly repeated spatial content.

It is important to visually compare the quality of the resulting images obtained with different transforms, Fig. 4 shows Lena image that is compressed/decompressed using the specified ICT-9 that is compared to the Def-ICT.

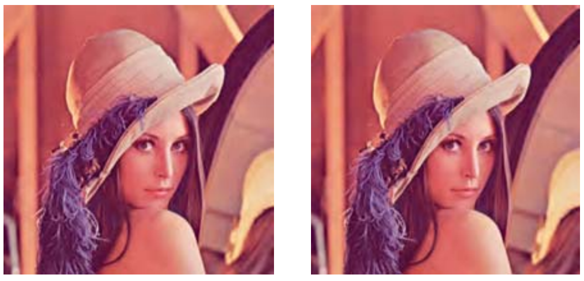

Fig 4: Showing the JPEG-2000 reconstructed Lena image after compression with rate 0.01. a) using *Def-ICT*, b) using *ICT-9*. The images above shows that the obtained ICT-9 has a good perceptual characteristics

The decision of which resulted ICT matrix best optimize the color transformation process for every possible image is a crucial, painstaking, and sensitive task. This is due to the different error values for each red, green, and blue band of each image which causes different estimated AUROC that cannot be compared wisely. The performance factor described below is used so that the promotion rate is the metric that designates the usefulness of each matrix. The metric outputs the improvement accomplished when using the GA-ICT over the default ICT matrix.

$$Performanc\ e\ Factor\ = \frac{Average\ [\mathrm{AUROC}\ _{\mathrm{Def\ -ICT}}]_{\mathrm{Re}\,d\,,Green\ ,Blue}}{Average\ [\mathrm{AUROC}\ _{\mathrm{GA\ -ICT}}]_{\mathrm{Re}\,d\,,Green\ ,Blue}} \times 100\,\% \tag{19}$$

The data presented in Table 6 discusses the Lena, Baboon, Peppers, Fractal and Texture performance respectively over all fitness functions, the matrix with maximum performance factor exhibits better performance. This is because large performance factor indicates little ROC area (i.e. RMSE) of the GA-ICT over the compression factor interval up to $C_{threshold-1}$ with respect to the ROC area of the default ICT. So *ICT-6*, *ICT-7*, *ICT-3*, *ICT-9*, and *ICT-10* performed the compression with highest image fidelity and least image distortion.

Table 6: Performance factor analysis

Image	Component	Default ICT	AUROC Fitness-1	AUROC Fitness-2	AUROC Fitness-3
Lena	**Red**	1.18514	1.02605	0.77414	0.76833
	Green	1.16270	0.86847	0.98165	1.09408
	Blue	1.58803	1.19870	1.17556	1.23721
	Average	1.31195	1.03107	0.97712	1.03321
	Performance Factor		**127.24 %**	**134.27 %**	**128.98%**
Baboon	**Red**	3.52015	3.24616	2.90878	3.24079
	Green	2.82948	3.44472	2.41681	3.62825
	Blue	3.99798	3.21172	3.84178	3.31504
	Average	3.44921	3.30087	3.05579	3.39469
	Performance Factor		**104.49 %**	**112.87 %**	**101.61%**
Peppers	**Red**	1.77235	1.37907	1.34073	1.49151
	Green	1.37565	1.50519	1.44303	1.18499
	Blue	1.82225	1.21378	1.33051	1.55241
	Average	1.65675	1.36601	1.37142	1.40964
	Performance Factor		**121.28 %**	**120.81 %**	**117.53%**
Fractal	**Red**	1.78012	1.57624	1.41356	1.51690
	Green	1.55325	1.55539	1.48992	1.44238
	Blue	1.93618	1.58752	1.65186	1.63580
	Average	1.75652	1.57305	1.51845	1.53169
	Performance Factor		**111.66 %**	**115.68 %**	**114.68%**
Texture	**Red**	4.30407	4.07748	3.59955	3.98080
	Green	3.29591	3.02464	3.63223	3.45145
	Blue	4.30732	3.90447	3.68704	3.73808
	Average	3.96910	3.66886	3.63961	3.72344
	Performance Factor		**108.18 %**	**109.05 %**	**106.60%**

Table 7: Universality property of each GA-ICT matrix

	Component	**AUROC**					
		Default ICT	**ICT-1**	**ICT-2**	**ICT-3**	**ICT-4**	**ICT-5**
Fitness-1	**Red**	2.51237	2.41266	2.32371	2.65029	2.22029	2.32897
	Green	2.04340	1.89545	2.42203	2.51828	2.15906	1.76411
	Blue	2.73036	2.41570	2.23425	2.47575	2.18195	2.34972
	Universality		**YES**	**NO**	**NO**	**NO**	**YES**
	Component	**Default ICT**	**ICT-6**	**ICT-7**	**ICT-8**	**ICT-9**	**ICT-10**
Fitness-2	**Red**	2.51237	2.84301	2.12971	2.39668	1.89280	2.05060
	Green	2.04340	2.44067	1.77252	2.68499	2.13041	2.14506
	Blue	2.73036	3.08124	2.63284	2.39965	2.35322	2.26694
	Universality		**NO**	**YES**	**NO**	**NO**	**NO**
	Component	**Default ICT**	**ICT-11**	**ICT-12**	**ICT-13**	**ICT-14**	**ICT-15**
Fitness-3	**Red**	2.51237	1.85368	2.36910	2.47448	2.14961	2.24637
	Green	2.04340	2.31300	2.60795	2.00614	1.99667	2.03608
	Blue	2.73036	2.50205	2.35548	2.63946	2.39531	2.29377
	Universality		**NO**	**NO**	**YES**	**YES**	**YES**

Table 8: Scoring of the UGA-ICT matrices

UGA-ICT	**Performance Factor**	**Condition Number Percentage**	**Standard Deviation Percentage**	**Score**
ICT-1	127.24 %	108.36 %	102.24%	112.61%
ICT-5	108.18 %	104.26 %	103.62%	105.35%
ICT-7	112.87 %	91.62 %	103.22%	102.57%
ICT-13	117.53%	121.47 %	99.75%	112.92%
ICT-14	114.68%	91.05 %	100.43%	102.05%
ICT-15	106.60%	150.57 %	104.23%	120.47%

The graph shown in Fig. 3 depicts the percentage of improvement in image fidelity for all images and under the three fitness functions used in this work. The figure reveals the superiority of fitness-function-2 in achieving better improvement for all images in most of the cases. Nonetheless, all GA-ICT matrices are better than the default ICT.

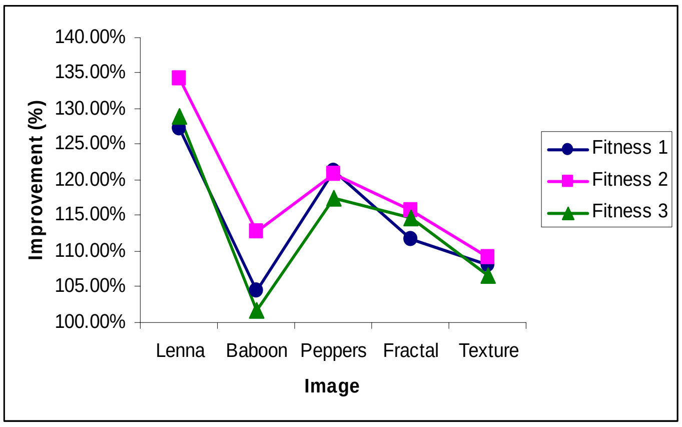

Figure 3: Performance improvement of each image

Hitherto, the performance analysis discussed the effectiveness of the resultant GA-ICT matrices for each image independent from its quality over the other images. Now, an estimation of the performance for each GA-ICT matrix considering its impact upon the other images will be performed. So, the universality of each matrix is weighted efficiently. The strategy adopted to estimate the quality of the GA-ICT depends on its superiority of all the red, green, and blue AUROC values. If any of these three AUROC values outweighs the default-ICT, it will indicate its deficiency in achieving less error value over the entire compression factor range and will not considered as universal. Equation (20) illustrates the formula of a universal GA-ICT (UGA-ICT). Table 7 elucidates the universality of each GA-ICT matrix.

$$UGA-ICT=\left\{\begin{array}{ll} GA-ICT(AUROC_{R,G,B}) \Big| & AUROC_R(ICT_{GA})<AUROC_R(ICT_{DEF}) \\ & ,AUROC_G(ICT_{GA})<AUROC_G(ICT_{DEF}) \\ & ,AUROC_B(ICT_{GA})<AUROC_B(ICT_{DEF}) \end{array}\right\} \tag{20}$$

As shown in Table 8, six *GA-ICT* matrices are considered as universal, each one of them can supersede the default-ICT used in the JPEG-2000 standard for the color transform intercomponent phase. To make use of the performance analysis cohesively, a score of the overall performance analysis can be employed, this employment is applied in Table 8. The score is the average of the *GA-ICT* values when compared to the default-ICT values. The performance factor of the GA-ICT, the improvement of the condition number improvement, and the standard deviation of the GA-ICT relative to the default ICT are the factors that shape this score. The scores in Table 8 verify that *ICT-15* is the most universal *GA-ICT* matrix, and it is a strong candidate to compete the performance of the default ICT and get better image quality with less distortion.

5 Conclusions

This work presents new color transformations based on using genetic algorithms and JPEG2000. The new color transformations which do not have the YC_bC_r characteristics have given improved results compared to the color transformations formally adopted by the JPEG2000 group. We recommend using *ICT*-9 as this work proves that it is the best color transformation. Also due to tiling used in JPEG-2000, the resultant GA-ICTs found in this work directly applies to larger or smaller images.

It must be noted that the YC_bC_r transform used in JPEG-2000 standard represents an image as its luminance component (for example Y = (R+G+B)/3) and as two opponent color components which are color differences. None of the transforms obtained in this work has such a structure of a luminance-chrominance transform. It is possible to introduce additional constraints in the algorithm to ensure that the resulting transform is an opponent color transform which is left as a future work. The obtained *ICT*s have better perceptual characteristics and this mean that the RGB decorrelation degree is higher than the original YC_bC_r. Since the found color transformations may replace the old ones, the computational complexity is not affected at all.

6 Acknowledgment

The authors of this work would like to thank Michael Adams for making the jasper source code publically available at www.ece.uvic.ca/˜mdadams.

7 References

[1] Skodras A., Christopoulos C., Ebrahimi T., "JPEG2000: The Upcoming Still Iimage Compression Standard", Pattern recognition letters, Vol.22, No.12, pp.1337-1345, 2001.

[2] Taubman D., Marcellin M., "JPEG2000: Standard For Interactive Imaging", Proceedings of IEEE, Vol.90, No.8, pp.1336-1357, 2002.

[3] ISO/IEC 15444-1, "Information Technology-JPEG2000 Image Coding System, Part 1, Core Coding System", Technical report, 2000.

[4] Ebrahimi F., Chamic M., Winkler S.,"JPEG vs. JPEG2000: An Objective Comparison Of Image Encoding Quality, SPIE Applications of Digital Image Processing, vol.5558, pp.300-308, 2004.

[5] Antonini M., Barlaud M., Mathieu P., Daubechies I., "Image Coding Using Wavelet Transform", IEEE transactions on image processing, vol.1, no.2, pp.205-220, 1992.

[6] Christopoulos C., Scodras A., Ebrahimi T.," The JPEG2000 Still Image Coding System: An Overview",IEEE transactions on consumer electronics, vol.46, no.4, pp.1103-1127, 2000.

[7] Acharya T., Tsai P.,"JPEG2000 Standard For Iimage Compression Concepts, Algorithms And VLSI Architectures", John Wiley and sons., 2005.

[8] CCIR Recommendation 601: Encoding Parameters Of Digital Television For Studio, Green book, 1990.

[9] Holland J., "Adaptation In Natural And Artificial Systems", University of Michigan Press, Ann Arbor, MI, USA, 1975.

[10] Parent J. and Nowe A., "Evolving Compression Preprocessors with Genetic Algorithms", In Proceedings of the GRONICS Student Conference, 2000.

[11] Vences L., Rudomin I., "Genetic Algorithms For Fractal Image Sequence Compression", Proceedings computaction visual, pp.35-44, 1997.

[12] Adams M., The JPEG-2000 Still Image Compression Standard, www.ece.uvic.ca/˜mdadams.

[13] Eskicioglu A., Fisher P., "Image Quality Measures And Their Performance, vol.43, no.12, pp.2959-2965, 1995.

[14] Kountchev R., Milanova M., Todorov V., Kountcheva R., Adaptive fuzzy filter for the reduction of blocking artifacts in images compressed with IDP decomposition and JPEG, WSEAS Transactions on Signal Processing, Vol. 2, No. 7, pp. 941-8, 2006.

[15] Idris F., Atef F., Bitplane overlap shift method for ROI coding in JPEG2000, WSEAS Transactions on Signal-Processing, Vol. 2, No. 5, pp.754-759, 2006.

[16] Dimitroulakos G., Galanis M.D., Goutis C. E., Optimized encoder architecture and filters for the tile-based convolutional 2D-DWT for the JPEG 2000 standard, WSEAS Transactions on Circuits and Systems, Vol. 5, No. 8, pp. 1342-1349, 2006.